AR-009-624

DSTO-RR-0074

The Autoscaling of Oblique Ionograms

Nicholas Redding

APPROVED FOR PUBLIC RELEASE

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

The Autoscaling of Oblique Ionograms

Nicholas Redding

Information Technology Division
Electronics and Surveillance Research Laboratory

DSTO-RR-0074

ABSTRACT

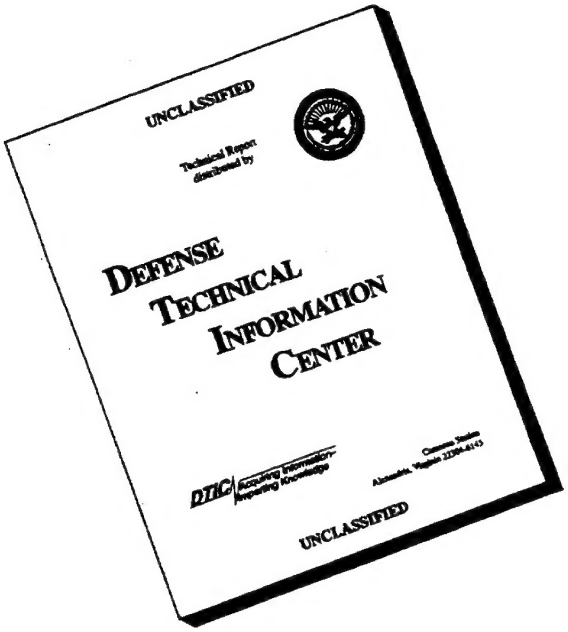
This report details the work done on developing an autoscaling system for the Low-Latitude Ionospheric Sounding Program (LLISP). The report firstly describes the performance of a number of different filtering routines for automatic cleaning of LLISP oblique ionograms. Secondly, it then presents techniques developed to automatically extract a feature vector from a filtered ionogram. A third stage uses the feature vectors to identify the prominent modes and other important features of the ionograms, and to track these modes and features as they evolve across sequences of ionograms. This work will automate the task of identifying and tracking ionospheric propagation modes in ionograms. The resulting system will be used to process current LLISP data streams to extract large volumes of significant information: this will help the understanding of ionospheric processes and can be used as input to automatic frequency management systems for communications networks.

APPROVED FOR PUBLIC RELEASE

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

Published by

DSTO Electronics and Surveillance Research Laboratory
PO Box 1500
Salisbury. South Australia, Australia 5108

Telephone: (08) 259 7053
Fax: (08) 259 5619

AR-009-624
March 1996

APPROVED FOR PUBLIC RELEASE

The Autoscaling of Oblique Ionograms

EXECUTIVE SUMMARY

The ionosphere is a region of partial ionization of the Earth's atmosphere extending from a height of about 100 km to more than 1000 km above the Earth's surface. Radio waves in the HF (High Frequency) band (roughly 3 to 30 MHz) are refracted through the ionosphere because of the presence of free electrons. The density of the refracting electrons varies with height, location above the earth, time of day, season, and intensity of solar radiation. This profile determines the propagation mode by which radio waves of a particular frequency are reflected by the ionosphere.

The reflecting property of the ionosphere mentioned above facilitates over-the-horizon communication in the HF band, but the complexity and the changing nature of the ionosphere makes the available frequencies and bandwidth difficult quantities to determine or predict. A single frequency can reach a receiver by a number of different modes which produce a distribution of time-delays from the transmitter to the receiver. *Ionograms* are used to determine the parameters for HF over-the-horizon communication systems. Each ionogram is a two-dimensional representation of the strength of a received test-signal for a range of frequencies and time-delays. In an *oblique* ionogram, the transmitter and receiver are separated by up to thousands of kilometers, and because the direct pathway from the transmitter to the receiver is obscured by the curvature of the earth, the signal reaches the receiver by bouncing off the ionosphere.

At present the information gained from this data is to a large degree qualitative: operators look at representations of individual ionograms as images and interpret interesting features in these images in terms of the underlying atmospheric physics. The sheer volume of this data, however, has saturated DSTO's capacity to make effective use of it all, creating the need for an *autoscaling* system that can automatically extract the salient information from raw ionograms, preferably in real time.

Thus autoscaling involves taking a raw ionogram, identifying its major features, and reducing these to a vector that usefully summarizes them. The system that is described here to do this has been developed in three overall stages. The first stage filters the raw ionograms to remove noise and enhance features, and this report describes the performance of a number of different filtering routines for automatic cleaning of ionograms. The idea behind feature extraction, the second overall stage, is to find a simple description of the traces in a symbolic form. This is achieved by finding a suitable model such that for some value of the model parameters there is a good correspondence between the shape of the trace and the shape of the model. This highlights a number of problems. Firstly, we need to choose a suitable model for the traces. Secondly, we have to find a measure of the goodness of fit and a fitting routine to determine a value of the model parameter that minimizes the measure. Lastly, we need to separate the pixels that correspond to particular traces and reduce them in number somehow because this will have a serious impact upon the overall speed of the autoscaling system (*i.e.,* the optimization involved in fitting is the rate determining step for speed in the autoscaling system). Incorporated into the feature extraction process is a stage designed to account for broken or missing features in an ionogram. *Merging* is the process of reconnecting trace fragments, which clearly belong to the same propagation mode, that have become disconnected for some reason (noise, attenuation, *etc.*) and so have been fitted separately. Thus merging is one more step designed to move towards the ideal where one trace corresponds to one propagation mode and vice versa.

The final overall stage of the autoscaling system includes, firstly, a tracking stage that is designed to assist mode identification in circumstances where one or more of the simple modes are missing from the ionogram because of various ionospheric effects; secondly, a mode identification stage which makes use of an ionospheric model to identify the propagation modes of each trace; and lastly a measurement stage to output the desired measurements from the ionogram.

This report finishes by examining the performance of the autoscaling system against 25 ionograms that were selected by a potential user of the system.

Author

Nicholas J. Redding
Information Technology Division

Nicholas Redding received a B.E. and Ph.D. in electrical engineering all from the University of Queensland, Brisbane, in 1986 and 1991, respectively. From 1988 he received a Research Scientist Fellowship from the Australian Defence Science and Technology Organisation (DSTO) and then joined DSTO in Adelaide as an Research Scientist after completing his Ph.D. in artificial neural networks in 1991. He was recently (1996) appointed as a Senior Research Scientist in the Microwave Radar Division of DSTO. Since joining DSTO in Adelaide, he has applied image processing techniques to problems in ionospheric physics. His current research interests include image processing, Gabor transforms and information retrieval.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The ionosphere is a region of partial ionization of the Earth's atmosphere extending from a height of about 100 km to more than 1000 km above the Earth's surface. Radio waves in the HF (High Frequency) band (roughly 3 to 30 MHz) are refracted through the ionosphere because of the presence of free electrons. The density of the refracting electrons varies with height, location above the earth, time of day, season, and intensity of solar radiation. This profile determines the propagation mode by which radio waves of a particular frequency are reflected by the ionosphere.

Propagation Modes The reflecting property of the ionosphere mentioned above facilitates over-the-horizon communication in the HF band, but the complexity and the changing nature of the ionosphere makes the available frequencies and bandwidth difficult quantities to determine or predict. A single frequency can reach a receiver by a number of different modes which produce a distribution of time-delays from the transmitter to the receiver. These modes can be "simple" modes, where the signal has bounced a number of times between the ground and the "F-region" of the ionosphere, where peak electron density occurs. The propagation modes may also be "complex" ones, where the signal has been reflected at some stage between less dense regions of the ionosphere and the ground, or even between the regions of the ionosphere itself.

Oblique Ionograms *Ionograms* are used to determine the parameters for HF over-the-horizon communication systems. Each ionogram is a two-dimensional representation of the strength of a received test-signal for a range of frequencies and time-delays. In an *oblique* ionogram, the transmitter and receiver are separated by up to thousands of kilometers, and because the direct pathway from the transmitter to the receiver is obscured by the curvature of the earth, the signal reaches the receiver by bouncing off the ionosphere. A signal at a particular frequency may reach the receiver by one, or more, possible pathways corresponding to the number of times it bounced off different regions of the ionosphere before reaching the receiver. Of course, the more a signal bounces off the ionosphere, the longer it will take to reach the receiver, so differing pathways produce differing time-delays for the signal. Figure 1.1 represents the pathway of a simple 2-hop propagation mode as a signal travels from a transmitter TX to a receiver RX.

LLISP Ionograms The oblique ionograms used here were produced under the Low-Latitude Ionospheric Sounding Program (LLISP) [1], which currently collects them as an ongoing commitment. At present the information gained from this data is to a large degree qualitative: operators look at representations of individual ionograms as images and interpret interesting features in these images in terms of the underlying atmospheric physics. The sheer volume of data from LLISP, however, has saturated DSTO's capacity to make effective use of it all, creating the need for an autoscaling system that can automatically extract the salient information from raw ionograms, preferably in real time. This report describes the work done to produce such a system under tasks DEF 88/Ionospheric Propagation and Reception Research and ADF 93/Real Time Frequency Management for HF Communications.

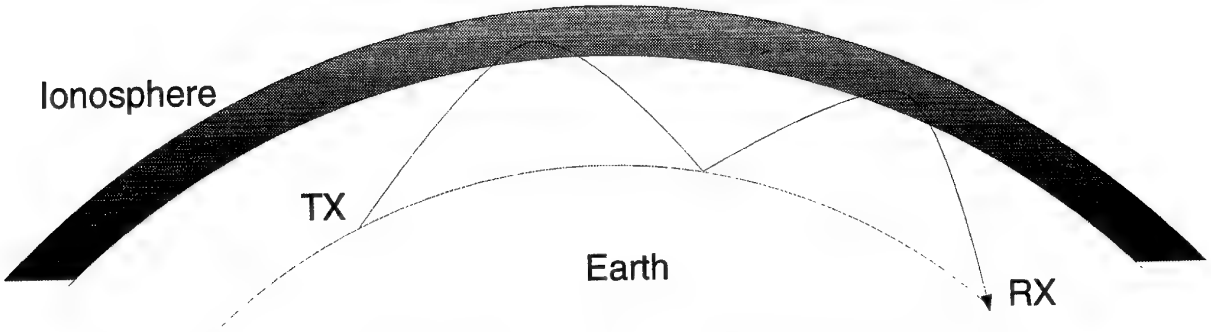


Figure 1.1: A simple 2-hop propagation mode of a signal as it travels from a transmitter TX to a receiver RX, where the receiver is obscured from the transmitter by the curvature of the earth.

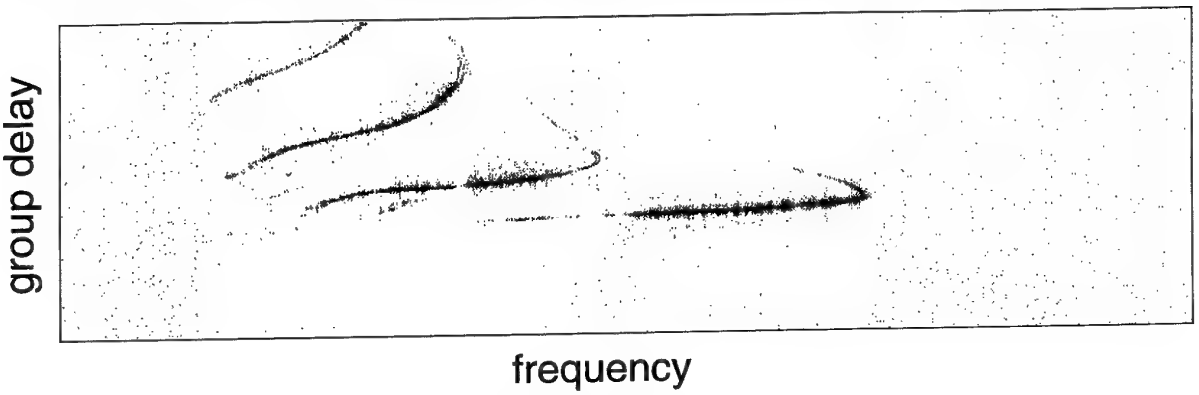


Figure 1.2: An example of an oblique ionogram.

Scaling and Autoscaling Ionograms are collected at regular intervals of from 5 minutes up to an hour. The oblique ionogram, which is presented to the operator as an image, has a number of characteristic traces that correspond to the propagation modes of the signal through the ionosphere. A number of measurements are currently made manually from these ionograms, and this is called *scaling*. These measurements range from simple band-availability measurements, where the presence or absence of a signal received at a particular frequency is determined, to a full characterization of the propagation modes that are currently possible through the ionosphere - these quantities change from minute to minute. The goal of this project has been to automate these measurements. The approach I have taken is to apply image processing techniques to the artificial images to the see what can be achieved by way of an *autoscaling* system.

Traces The most prominent features in an ionogram are the *traces*. These are connected segments in the ionogram and reflect discrete structures and associated modes of signal propagation in the ionosphere: they are clearly visible in figure 1.2. The traces have definite physical interpretations — each trace is produced by a range of frequencies that have reached the receiver by a single mode of propagation. The ionogram and others seen latter in this report, also shows that the traces and associated modes can be obscured by noise and other effects, that they can be broken with several distinct traces being obviously associated with the one underlying structure, and that they can take quite complicated forms. Furthermore, we will see latter that traces due to more than one mode of propagation can be joined into a single complicated trace structure. Thus the primary aim of the work described here has been to isolate the most significant trace-like features, and to then perform measurements on them.

In the following discourse, I will refer to a trace as a *trace structure* when I want to emphasize that the trace contains signal that was received by more than one ionospheric propagation mode, and as a *trace fragment* or *trace segment* when I want to emphasize that it is only a small part of the signal received by a single propagation mode. Ideally when I use the term trace, I am referring to all the signal that corresponds to a single propagation mode and no other, and much of the processing in this autoscaling

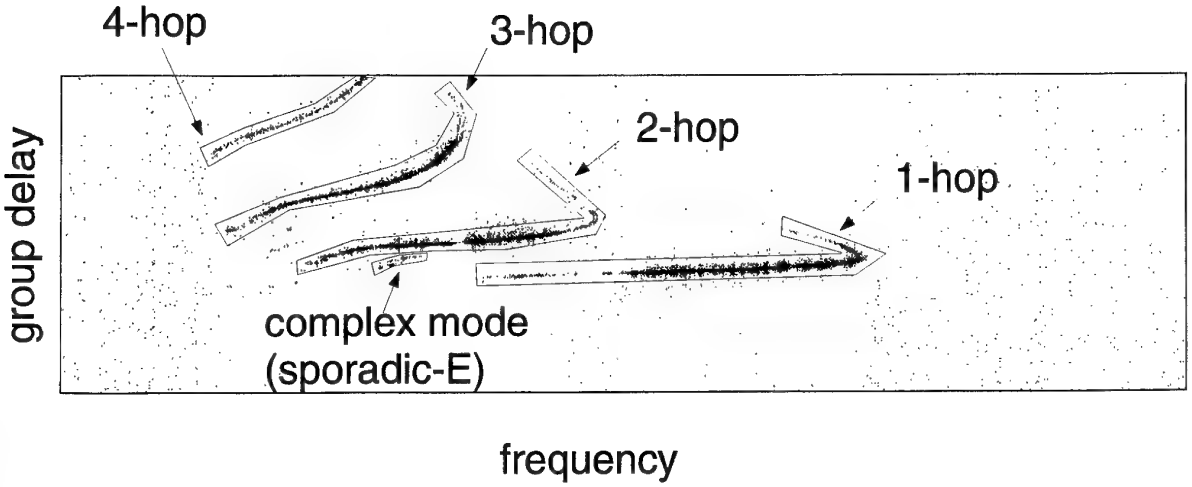


Figure 1.3: A typical ionogram with the information determined by an autoscaling system annotated.

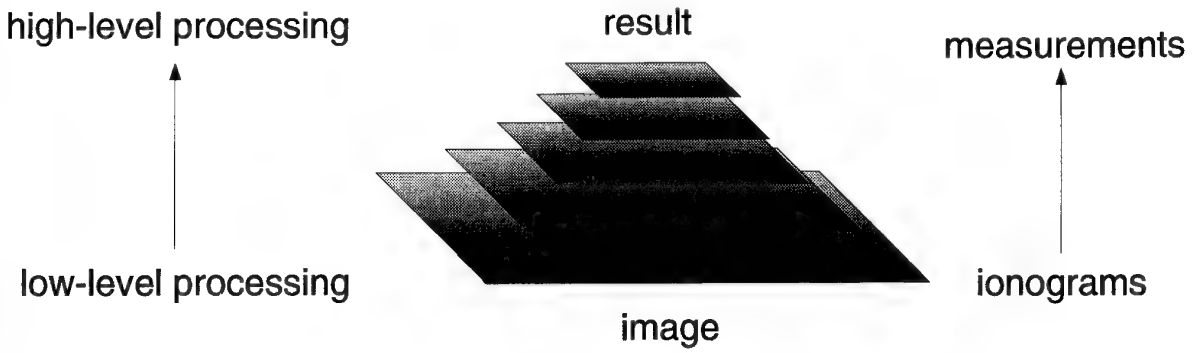


Figure 1.4: The processing pyramid: each layer of the pyramid corresponds to a step in the processing scheme, where the area of the layer is represents the quantity of data at each stage.

system is to try and achieve this ideal.

The typical ionogram of figure 1.2 has been annotated with some of the sort of labels that need to be determined by the autoscaling system in figure 1.3. The propagation modes of individual traces in the ionogram, and the strengths of these modes, are used to derived parameters which characterize the ionosphere at a particular point in space and time. Therefore, in order to solve the autoscaling problem, we need to be able to separate out the traces that correspond to a particular propagation mode, and identify what that propagation mode is.

Principles of Processing Ideally an autoscaling system would produce useful quantitative summaries of ionograms by combining knowledge of the underlying ionospheric physics with well-defined requirements. Unfortunately, uncertainty about the about the contributing phenomena and the complexity of the physical models makes this approach prohibitive from both a technical and a computational perspective. Therefore the approach taken here is to produce a system that emulates the current actions of operators who manually scale visual representations of ionograms primarily by simple pattern recognition. The difficulty is, however, that while it is easy for the human eye to follow trends and extrapolate across breaks to precisely determine whether trace segments should be joined, the task is not so easy to implement as a computer algorithm.

In general terms, the steps involved in any image processing system can be represented symbolically by a pyramid (see figure 1.4). Each higher layer of the pyramid represents a successive step in the processing scheme. In each step the amount of data under consideration (represented by the area of each level of the pyramid) is reduced as it becomes increasingly more abstract. So at the first stage the

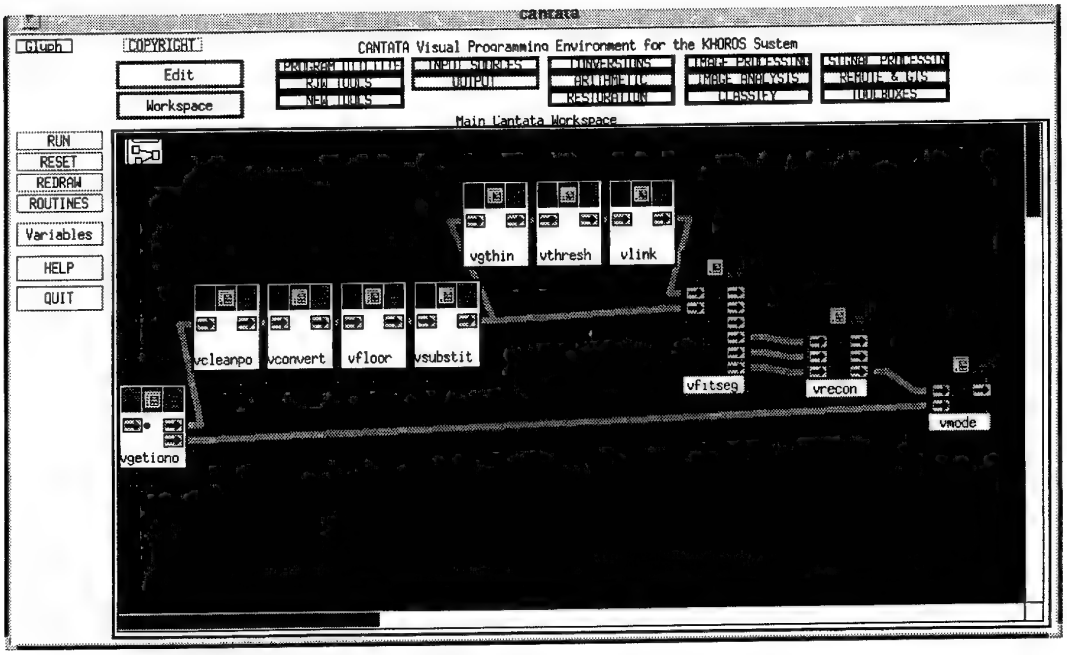


Figure 1.5: The Khoros visual program for the autoscaling system.

scheme manipulates all the pixels of the image, while in the final stages only derived quantities are dealt with until the desired measurements are obtained.

Thus autoscaling involves taking a raw ionogram, identifying its major features, and reducing these to a vector that usefully summarizes them. The system that is described here to do this has been developed in three overall stages. The first stage filters the raw ionograms to remove noise and enhance features, and the next chapter of this report describes the performance of a number of different filtering routines for automatic cleaning of ionograms, and evaluates their performance on the task of removing the noise from oblique ionograms. This chapter also briefly considers some of the implications of the results obtained for the further definition of the overall system.

Next, Chapters 3 and 4 reports on the second stage of the overall system which isolates each feature and finds an effective summary representation for it in a process called *feature extraction*. Chapter 5 details how these features are refined to account for broken and missing traces by merging features as appropriate. Mode identification and measurement stage is documented in Chapter 6 and forms the last stage of the autoscaling system. This covers tracking features as they evolve across sequences of ionograms to account for missing features. The last chapter has a performance assessment of the existing system.

Implementation The system is implemented under Khoros [2], a data flow visual language system for image processing that allows the user to write additional modules for special functions in the C language. The Khoros visual program for the autoscaling system is depicted in figure 1.5. Many image processing routines are available under Khoros, but the majority of the routines that comprise the autoscaling system had to be specifically written for it. The Khoros system has proven to be an excellent research platform for development of the autoscaling system in part because of the many pre-existing routines it offers. Although only one significant routine formed part of the final solution, it did allow me to eliminate those provided quickly from consideration. It is also valuable because it allows modules to be added and easily interconnected in any order and because the processing stream to be easily visualized. However, the final production version of the autoscaling system should be implemented in a manner that does not require any of Khoros, because in a production system the functionality that Khoros provides would serve no purpose and be an unnecessary burden.

Philosophy The terms used by ionospheric physicists to characterize an ionogram are phenomenological and descriptive — features in an ionogram are characterized in terms of the physical process that

produces them and is just as much qualitative as it is quantitative. The number and type of modes and their prominent structural features are emphasized. The result is a scaling that describes the ionogram in terms of its causes and processes rather than of what its implications are for other activities.

In contrast, communications engineers appear to be more interested in quantitative application-based descriptions, such as measures of availability of particular frequency bands. The result of this emphasis is a characterization of the ionogram in terms of its implications rather than its causes or internal structure.

In either case, gathering useful information from large numbers of ionograms is only possible if there is an automatic way of producing a useful quantitative description of each individual ionogram. It is probably much easier to produce such summary statistics for particular application-based scalings than it would be for the process-based scalings. Nevertheless, it is the more ambitious automation of the process-based scalings that has been attempted here because of its possible general application.

Musings Finally, the system presented here did not spring complete like Athena from the author's mind. Rather it is the result of many trials and discussions: in particular a number of different approaches were explored in detail and turned out to be blind alleys. A particular problem has been finding representations of traces that are robust and can be easily calculated. This is an essential practical requirement, but one that may not be reconcilable with the goal of obtaining representations whose parameters have natural interpretations in the language of atmospheric physics. Indeed the aim of this paper is not only to report the work done, but is also to get further feedback from users as to the appropriateness of the isolation and representation procedures so that they can be suitably refined in any further work. In particular, while the focus here has been on isolating and describing traces, there may be other summary statistics or descriptors that should also be produced by an autoscaling system.

Chapter 2

Filtering

The problem addressed in this chapter is the removal of "noise" from an ionogram. Noise in an ionogram is a difficult quantity to define, and the definition I have used here may need refining. As a working definition, I have defined noise to be those data points of an ionogram that clearly do not compose part of a trace or trace fragment [1]. The actual ionospheric transmission mode that corresponds to each trace or trace fragment does not affect the filtering process and so will not be considered in this chapter. Identification of the transmission mode of each trace will be made by succeeding stages of the proposed autoscaling system, and will be discussed later.

To test the various filtering procedures, I have selected four ionograms that are representative of the variety of oblique ionograms that I have observed, at least with regard to those aspects that are challenging to a filtering system. The algorithms tested here include a variety of median filters, quantile filters, contour smoothing via nonlinear diffusion, a Crimmin's speckle filter, and a polynomial-fit cleaning filter. A number of other filters were tested and rejected for their poor performance; they included statistical filters based on such statistics as mean, variance, entropy and dispersion, and spatial frequency domain filtering, and convolution operations with a variety of different kernels.

A discussion of the important aspects of the four selected test ionograms follows in the next section. This is followed by an outline of the basic types of noise in imagery. Section 2.3 presents a brief introduction to the filtering algorithms, while section 2.4 describes how these algorithms were utilized in the filtering procedures. The results of the filtering procedures are presented in section 2.5; this is followed by a section of discussion and recommendations. The figures at the end of the chapter include all the test ionograms as filtered by the different procedures.

2.1 Test Ionograms

Four representative LLISP oblique ionograms were selected that had all the features I considered would be problematic for any filtering algorithm. Therefore, any filtering algorithm that performs well on these ionograms should also perform well on the majority of oblique ionograms. The four selected ionograms are shown in figures 2.1–2.4.

The first ionogram, shown in figure 2.1, has a number of ill-defined traces that are difficult to distinguish from the noise. These diffuse regions could easily be eliminated as noise by an over-enthusiastic filtering algorithm, and it is challenging to ensure that they are not removed. This ionogram has the additional problem that the traces due to one propagation mode are not completely connected.

The second ionogram in figure 2.2 presents the same difficulties as the first, along with a high level of clutter from the high adjacency of the complex-mode traces.

The third ionogram (figure 2.3) also has connectivity problems in the traces, a small number of trace fragments from complex modes, and a diffuse region at the maximum observed frequency.

The last ionogram, depicted in figure 2.4, again has connectivity problems, trace fragments, herringbone striations, and magneto-ionic splitting (the "forked tongue" effect) in the high-angle ray of two of the simple modes.

For testing purposes, the four ionograms were filtered by hand so that only those pixels deemed to be part of the signal were present. The four hand-filtered ionograms are shown in figures 2.5–2.8.

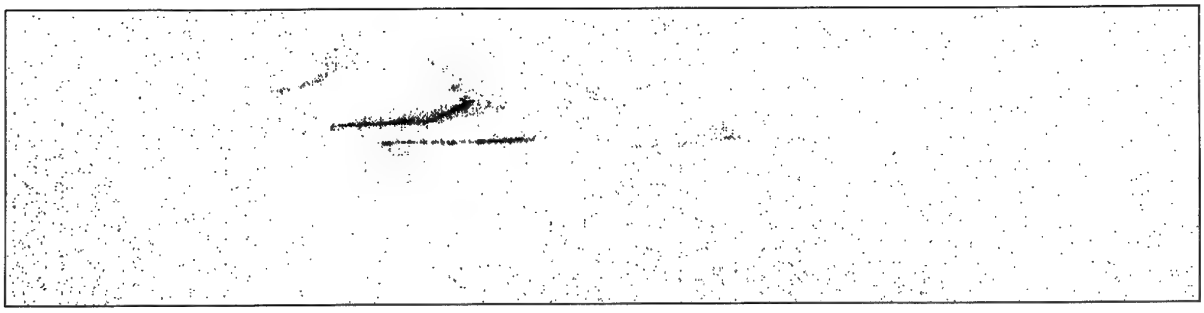


Figure 2.1: Ionogram 1 — original.

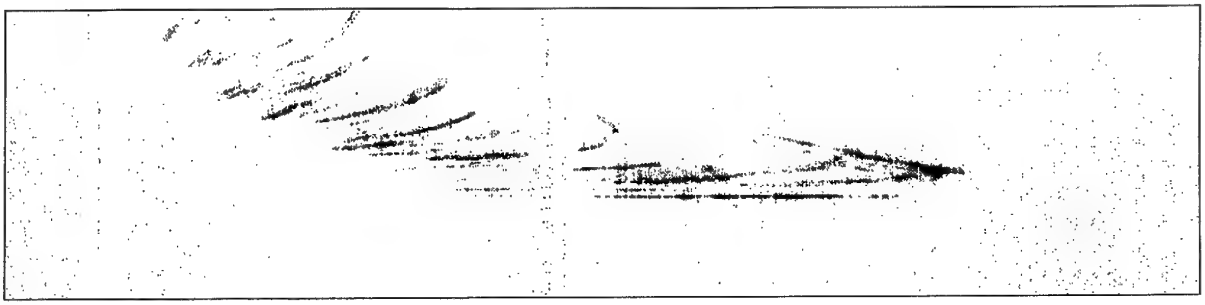


Figure 2.2: Ionogram 2 — original.

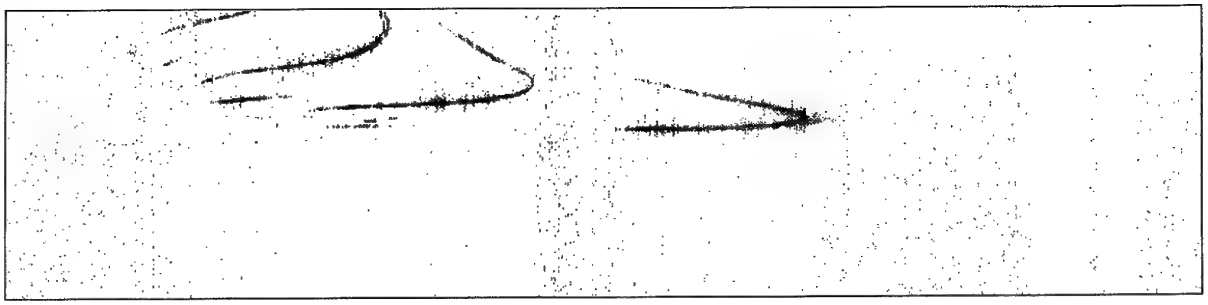


Figure 2.3: Ionogram 3 — original.

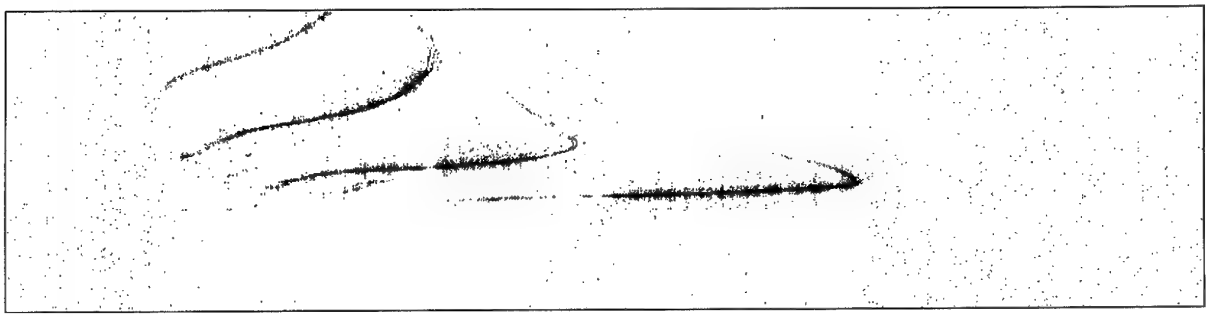


Figure 2.4: Ionogram 4 — original.

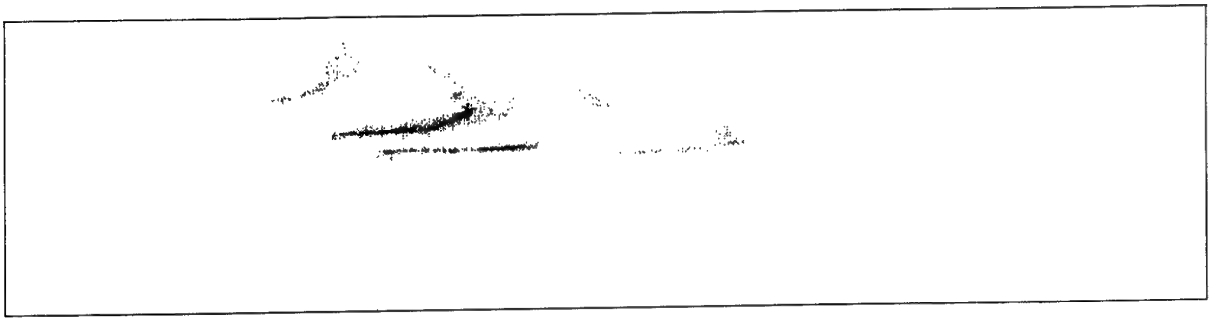


Figure 2.5: Ionogram 1 — hand filtered.

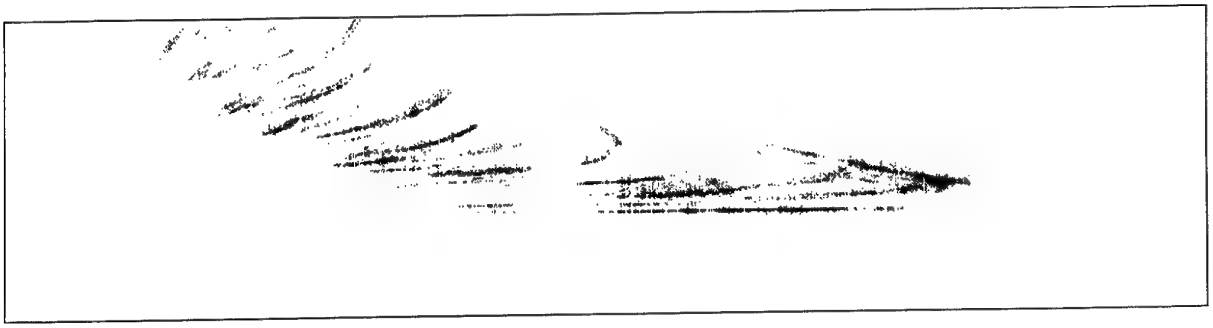


Figure 2.6: Ionogram 2 — hand filtered.

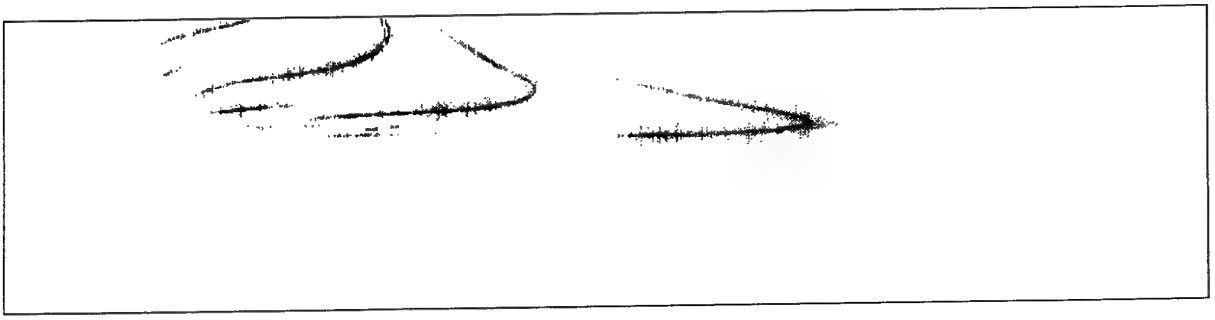


Figure 2.7: Ionogram 3 — hand filtered.

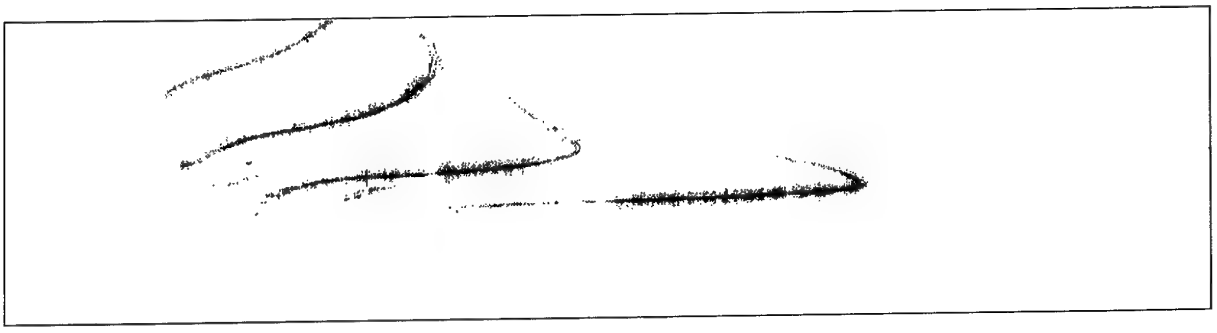


Figure 2.8: Ionogram 4 — hand filtered.

2.2 Noise Models

There are three basic types of noise that will concern us: additive noise, multiplicative noise and impulsive noise. Additive noise is modelled by

$$v(x,y) = u(x,y) + N(x,y)$$

where $u(x,y)$ is the image, and $N(x,y)$ is the noise distribution (which is commonly taken to be Gaussian and can be correlated from pixel to pixel). Multiplicative noise or speckle is modelled by

$$v(x,y) = u(x,y) \cdot N(x,y)$$

where N is usually taken to be exponentially distributed with mean 1. The third type of noise that will interest us is called *shot* or *impulsive* noise. In this model, a collection S of the pixels of an image are selected (via some distribution) for replacement by a value which may be dictated by a second distribution. That is:

$$v(x,y) = \begin{cases} N(x,y) & \text{if } (x,y) \in S(x,y) \\ u(x,y) & \text{otherwise.} \end{cases}$$

Again N is usually taken to have an exponential or Poisson distribution.

2.3 Algorithms

A brief description of each of the various filtering algorithms trialed on the ionogram data follows.

2.3.1 Median Filter

The median filter, a *rank* filter, is the simplest of the filters tested. This filter replaces each pixel in the image by the median of the pixels contained in a window around the pixel. That is,

$$v(x,y) = \text{median}(u(x-i, y-j) : (i,j) \in W_{mn})$$

where $u(x,y)$ and $v(x,y)$ are the unfiltered and filtered images, respectively, and $W_{2k+1,2l+1} = \{-k,\ldots,0,\ldots,k\} \times \{-l,\ldots,0,\ldots,l\}$ for a $(2k+1) \times (2l+1)$ window. The principles of the windowing operation of a median filter is depicted in figure 2.9.

The median filter performs reasonably well on additive non-Gaussian, impulsive, and multiplicative noise.

2.3.2 Quantile Filter

The quantile filter is a straight-forward generalization of the median filter. It replaces each pixel in the image by the q-th quantile of the pixels contained in a window around the particular pixel. The quantile of a set of p values, with parameter q, $0 \leqslant q \leqslant 1$, is defined to be the value that lies at position $[1 + q(n-1)]$ in the list of values sorted in ascending order, where $[\cdot]$ denotes the rounding operation. That is,

$$v(x,y) = \underset{q}{\text{quantile}}(u(x-i, y-j) : (i,j) \in W_{mn})$$

where the notation is defined as above.

The quantile filter performs reasonably well on additive non-Gaussian, impulsive, and multiplicative noise when the signal is rather faint.

2.3.3 Mean Filter

The mean filter replaces each image pixel by the mean of the pixels contained in a window around the pixel, *i.e.*,

$$v(x,y) = \text{mean}(u(x-i, y-j) : (i,j) \in W_{mn}).$$

The mean filter is a linear filter and performs well in situations where the noise is additive Gaussian.

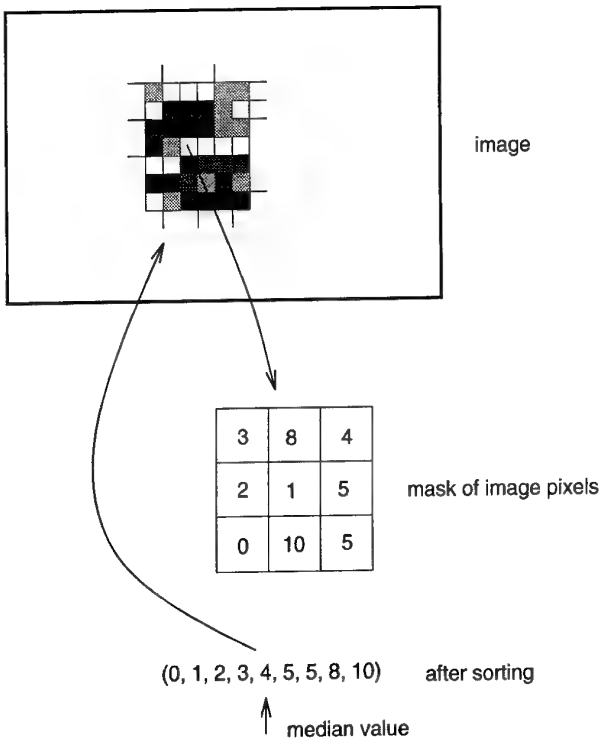


Figure 2.9: In the median filter, a *rank* filter, the window of pixel values around any particular pixel is sorted into ascending order, and then the median of this list becomes the new pixel value.

2.3.4 Crimmin's Speckle Filter

The Crimmin's speckle filter [3,4] is a nonlinear iterative filter based on geometric concepts, designed to reduce speckle from imagery. It has the property that narrow spines protruding from objects will be reduced, whilst wide spines are preserved. Similarly, narrow valleys are filled in whilst wide valleys are preserved.

2.3.5 Contour Smoothing via Nonlinear Diffusion

The contour smoothing algorithm tested here is a fast approximate version [5] of a nonlinear diffusion technique proposed in [6]. One iteration of the algorithm deforms the image according to the partial differential equation

$$\frac{\partial u(x,y)}{\partial t} = |\nabla u(x,y)| \, \nabla \cdot \left(\frac{\nabla u(x,y)}{|\nabla u(x,y)|} \right).$$

Essentially, this PDE says that the change in a particular pixel $u(x,y)$ is proportional to the product of the slope at that point and the second derivative of the image along the tangent to the gradient at (x,y). In other words, sharp turns in the contours of the image will be straightened out at a rate proportional to the slope across the contours. This means that strong straight edge features will be preserved, but localized variations will be smoothed. Good speed performance is achieved in the algorithm by assuming that all contours are locally circular [5].

The locally circular assumption is utilized in the following way. Let us assume that in some neighbourhood $\mathbb{B}$ the contours are circular with centre at the origin. Then, in this neighbourhood $\mathbb{B}$, the image can be written as a circularly symmetric function

$$u(x,y) = g(\sqrt{x^2 + y^2})$$

so then

$$\frac{\partial u(x,y)}{\partial t} = |\nabla u(x,y)| \, \nabla \cdot \left(\frac{\nabla u(x,y)}{|\nabla u(x,y)|} \right)$$

$$= \frac{g'(\sqrt{x^2+y^2})}{\sqrt{x^2+y^2}}$$
$$= \frac{g'(r)}{r} \tag{2.1}$$

where r is the radius of curvature of the contour. The PDE in (2.1) has solutions $g(t+\frac{1}{2}r^2)$. Therefore, the effect of (2.1) is to move each locally circular contour section towards its centre of curvature, reducing its radius of curvature.

The algorithm is thus implemented by estimating the centre of curvature for each pixel at every iteration. This centre of curvature is used to locate a point in the image whose value moves to the current pixel in one time step. The filtered value of the current pixel is found by a second order multinomial interpolation.

2.3.6 Filtering via Polynomial Fitting

This unpublished filter, developed by Whatmough [5], fits a second order multinomial in two variables to each 3×3 window in the image. It assumes that the highest order coefficients of the multinomial are only influenced by noise in the image and therefore these coefficients can be used to estimate the noise in the image at each window position. The filtered value is given by a mean of the raw centre value and the window mean, weighted according to the signal and total power. This filter can thus be considered to be an adaptive mean filter.

The algorithm works in the following way. Let $\frac{1}{\sqrt{3}}[1,1,1]$, $\frac{1}{\sqrt{2}}[-1,0,1]$, and $\frac{1}{\sqrt{6}}[1,-2,1]$ be the orthonormalized output of the functions $1, x, x^2$ on the points $x \in \{-1,0,1\}$. Let these three orthonormal functions be denoted by $f_0(x)$, $f_1(x)$, and $f_2(x)$, respectively. Thus in 2-D, the orthonormal multinomials (up to quadratic $\times$ quadratic) are

$$f_{ij}(x,y) = f_i(x) \cdot f_j(y), \;\; x,y \in \{-1,0,1\}, \;\; i,j \in \{0,1,2\}.$$

Next, each 3×3 window of the image is fitted to the second order orthonormalized multinomials just described. Therefore, a pixel $u(x,y)$ can be written as

$$u(x,y) = \sum_{p=0}^{2}\sum_{q=0}^{2} c_{pq} f_{pq}(x,y)$$

where

$$c_{pq} = \sum_{m=-1}^{1}\sum_{n=-1}^{1} f_{pq}(m,n)u(x+m,y+n)$$

because the functions $f_{pq}(x,y)$ form a basis.

The following assumptions are now made:

- c_{00} is proportional to the mean grey level of the image around (x,y).
- c_{10}, c_{01}, c_{20}, c_{11}, c_{02} reflect the local image detail and are also subject to noise, which is assumed to be additive Gaussian. Therefore,

  $$E(c_{10}^2) = \bar{c}_{10}^2 + \sigma_n^2$$

  and similarly for c_{01}, c_{20}, c_{11}, c_{02}, where $\bar{c}_{10}$ is the value of c_{10} with no noise, and σ_n^2 is the noise variance.
- c_{21}, c_{12}, c_{22} reflect only the presence of noise. Therefore,

  $$E(c_{21}^2) = E(c_{12}^2) = E(c_{22}^2) = \sigma_n^2.$$

From these assumptions, the following estimates about the non-DC component of the signal power S and noise power N can be made:

$$\begin{aligned} N &= \frac{8}{3}(c_{21}^2 + c_{12}^2 + c_{22}^2) \\ S &= (c_{10}^2 + c_{01}^2 + c_{20}^2 + c_{11}^2 + c_{02}^2) - \frac{5}{3}(c_{21}^2 + c_{12}^2 + c_{22}^2). \end{aligned}$$

If $S < 0$ by the above estimate then set $S = 0$. Let the mean signal in the 3×3 window around (x, y) be

$$\mu(x, y) = \frac{1}{9} \sum_{m=-1}^{1} \sum_{n=-1}^{1} u(x + m, y + n)$$

Then, in the spirit of Wiener filtering, let the new pixel value $v(x, y)$ be

$$v(x, y) = \frac{Su(x, y) + N\mu(x, y)}{S + N}$$

so that $v(x, y) \simeq u(x, y)$ when $S \gg N$ and $v(x, y) \simeq \mu(x, y)$ when $S \ll N$.

This algorithm seems to be a special case of the least mean squared error estimation algorithm described in [7] and is both robust and optimal [8]. The algorithm, because it is a linear regression estimation algorithm that employs normalized orthogonal basis functions, is inherently well-conditioned and will perform well even if the data is coarsely quantized (as indeed we shall see).

2.4 Procedures

In this section I detail how the algorithms outlined in the previous section were utilized in filtering the noise from the ionograms. The seven best procedures are reported here, and only for the best choice of of window dimension, parameter values and iteration counts. Note that all ionograms were normalized to have a peak signal strength of 255 before filtering was carried out.

Simple Median Filter

The median filter employed had a 3×3 window, and is principally included amongst these procedures as a benchmark.

Combined Median Filters

The combined median filter procedure used the output of three median filters in an effort to retain more of the signal in the filter output than using a single 3×3 median filter allowed. The window width $\times$ height of these three filters were 1×9, 11×1 and 3×3. The motivating idea was that the long horizontal and vertical windows would retain more of the faint traces (along the horizontal and vertical axes) in the filter output. The results of these three filters were combined by first taking the output of the 1×9, except where it was zero ir which case the value of 11×1 was used, except where it was also zero, in which case the result of the 3×3 window gave the pixel result.

Small Quantile Filter

A quantile filter with a 3×3 window was used with $q = 0.75$ and then the output was rescaled to 255. The value of $q = 0.75$ was visually determined to be the best for a 3×3 window.

Larger Quantile Filter

A quantile filter with a 5×5 window was used with $q = 0.8$ and then the output was rescaled to 255. This resulted in more smoothing and the retention of fainter structure in the filter output than in the case of the smaller quantile filter.

Mean and Speckle Filter

Two filters were used in series for this filter procedure. Firstly, a mean filter with a 3×3 window was applied to the image to smooth out the signal and then the output was rescaled to 255. Then, to remove the speckle noise, a Crimmin's speckle filter was applied to the image for one iteration. Finally, to remove noise residual from the background, all values below 20 were clipped from the ionogram.

Convolution and Contour Smoothing

Two filters were used in series for this filter procedure. Firstly, the image was convolved with a circle of diameter 5 to reduce the shot noise in the ionogram. Next, after rescaled to 255, the ionogram was filtered for 10 iterations using the contour smoothing filter. Finally, to clean up the background, all values below 10 were clipped from the image.

Polynomial-Fit Filtering

The polynomial-fit cleaning filter was applied to the ionogram for 10 iterations and the background cleaned by clipping all values below 20 after first being rescaled to 255.

2.5 Results

Eight separate statistics were gathered for each of the seven procedures on all four test ionograms. These statistics were

1. The percentage of signal pixels accepted by the filtering procedure, as measured against the hand filtered ionograms.

2. The percentage of noise pixels rejected from the ionogram by the filtering procedure.

3. The percentage of the background pixels (a zero value in the original image) that became signal pixel in the filtered ionogram. This is indicative of the amount of smoothing carried out by the filtering procedure.

   The following statistics are of lesser interest. They were gathered mainly as a "sanity check" on the filter output to ensure that nothing pathological had happened. A small value in each of these tests indicates a good result.

4. The magnitude decrease (as a percentage) in those signal pixels which have a filtered value less than the original.

5. The magnitude increase (as a percentage) in those signal pixels which have a filtered value greater than the original.

6. The magnitude decrease (as a percentage) in those noise pixels which have a filtered value less than the original.

7. The magnitude decrease (as a percentage) in those noise pixels which have a filtered value greater than the original.

8. The amount that the background pixels have increased in magnitude, expressed as a percentage of the summed magnitude of the true signal pixels. This statistic will also be indicative of the amount of smoothing in the filtered ionogram.

The results of the seven filtering procedures are reported in table 2.1.

Table 2.1: Filter Results

Algorithm	Image	Signal Accepted	Noise Rejected	Background as Signal	Signal Magnitude Dec.	Signal Magnitude Inc.	Noise Magnitude Dec.	Noise Magnitude Inc.	Background Mag. Inc.
3x3 median	1	60.26	99.83	0.09	0.22	0.05	0.61	0.00	0.16
	2	69.09	99.30	0.27	0.14	0.04	0.26	0.00	0.46
	3	76.13	99.61	0.19	0.62	0.10	1.31	0.00	0.61
	4	73.83	99.82	0.26	0.30	0.11	0.64	0.00	0.45
Combined median	1	73.11	99.76	0.15	0.08	0.00	0.61	0.00	0.19
	2	81.71	96.08	0.65	0.19	0.05	0.26	0.00	1.76
	3	87.05	98.09	0.32	0.45	0.10	1.18	0.00	0.97
	4	85.09	98.89	0.42	0.30	0.09	0.65	0.00	0.79
Quantile 3×3 $q = 0.75$	1	82.39	92.73	0.44	0.18	0.10	0.58	0.05	1.64
	2	90.21	84.03	1.35	0.04	0.38	0.23	0.00	4.04
	3	91.37	88.06	0.84	0.64	1.19	2.31	0.21	3.42
	4	90.42	91.03	0.94	0.11	0.70	0.92	0.01	2.38
Quantile 5×5 $q = 0.8$	1	80.55	99.63	0.50	0.13	0.16	0.61	0.00	3.94
	2	88.92	96.02	2.05	0.05	0.50	0.26	0.00	8.56
	3	91.61	97.66	1.15	0.36	1.58	2.63	0.13	8.04
	4	91.72	98.71	1.33	0.06	0.87	0.79	0.05	3.70
Mean 3×3 + speckle + clipping	1	94.35	61.81	1.80	0.28	0.00	0.99	0.00	3.57
	2	96.33	85.54	2.28	0.88	0.08	0.90	0.00	5.00
	3	97.45	66.27	1.73	2.09	0.64	3.24	0.00	5.64
	4	97.06	85.21	1.64	0.91	0.17	1.12	0.00	2.72
Convolution w. circle + contour smoothing	1	91.49	94.66	1.36	0.28	0.00	0.61	0.00	4.32
	2	95.37	92.85	3.88	1.28	0.11	2.70	0.00	8.77
	3	96.92	91.19	2.40	4.70	0.81	10.53	0.00	8.93
	4	97.13	94.54	2.74	1.59	0.36	3.16	0.00	4.18
Polynomial-fit cleaning + clipping	1	88.99	89.59	0.69	0.28	0.00	0.61	0.00	1.74
	2	94.52	94.68	1.91	0.27	0.18	0.37	0.00	4.11
	3	95.41	90.23	1.08	1.57	0.89	2.72	0.00	4.37
	4	95.58	96.31	1.27	0.71	0.26	0.89	0.00	2.05

2.6 Discussion and Recommendations

2.6.1 Objective Comments

By the quantitative results in table 2.1, the best performer is the convolution/contour smoothing procedure. The polynomial-fit filtering with clipping procedure is not quite as good a performer but is faster in execution. If speed is critical, the 5×5 quantile filter performs nearly as well with a very fast execution time.

Both the 5×5 quantile filter and the convolution/contour smoothing procedure have a high smoothing property as indicated by the higher values of the background magnitude increase percentages, and this is borne out when one examines figures 2.23–2.26 and figures 2.31–2.34. The best performer, the convolution/contour smoothing procedure, has some decrease in the signal magnitude (particularly on ionogram 3 at 4.70 %), but this is not a problem when compared with the much larger noise magnitude decrease. None of the techniques trialed display any notable increase in the magnitude of the noise still present after filtering.

The 28 filter results are presented in figures 2.11–2.38 in the appendix. Note that the printing process has emphasized the low-valued pixels so that the noise background appears stronger than its actual magnitude.

A number of further observations can be made after examining the filter outputs in figures 2.11–2.38:

- The median filter is over-enthusiastic at removing noise. It leaves the traces very fragmented.
- The combined median filter leaves a number of vertical and horizontal artifacts.
- The smoothing in the 5×5 quantile filter is much higher than in the 3×3. For this reason one may argue that the 3×3 better reflects reality, depending on the resolution of the trace-truth problem.
- The polynomial-fit cleaning procedure is the second-best performer by table 2.1 and has a low level of smoothing in figures 2.35–2.38.

2.6.2 Chosen Filter

In addition to the quantitative tests of table 2.1, there are two issues hinted at above in the objective comments that have dictated the final choice of filtering procedure for the autoscaling system. Firstly, the autoscaling system must process many ionograms and needs to be as fast as is reasonable, and for this reason the convolution/contour smoothing algorithm is an undesirable choice. Secondly, I have noted that any filtering procedure that produces a lot of smoothing in the ionogram is more likely to erroneously connect traces segments that should not be connected (*i.e.*, those that are due to different propagation modes). This is important, because in the next processing stage of the autoscaling system, the feature extraction stage, traces are defined to be contiguous regions of connected pixels. The polynomial-fit cleaning procedure is a good compromise solution for all these problems: it is the second-best performer by table 2.1; it is reasonably fast, although not as fast as the quantile filter; and qualitatively it has an acceptable level of smoothing which is lower than that of the convolution/contour smoothing procedure.

2.6.3 Issues of Interpretation

During the process of hand-filtering, all pixels in the region of what appeared to be trace fragments were assumed to be signal and were left untouched. This will not be true if multiplicative or additive noise is present. A more precise assessment by an expert of noise and signal would be helpful. For example, in [1] it is noted that the herring-bone striations are an artifact of the method of ionogram formation. Is, then, the true ionogram trace a smooth ridge formed by trimming off the striations or is the true trace a thickened ridge formed by smoothing out the energy in the striations along the ridge? Accurate measurements will require these questions to be resolved. It seems however, that this is a very ill-defined problem.

It may well be that an expert assessment would dictate a different choice of winning filtering algorithm than the one I have chosen based on the quantitative results of table 2.1. For example, it may be that figures 2.31–2.34 indicate too much smoothing, resulting in traces that are broader than reality. This problem may be suitably titled the *trace-truth* problem by analogy with the ground-truth problem of Remote Sensing.

Using my own assessment of "truth" (that portion of the ionogram deemed to be signal) we can examine the noise and signal distributions. An examination of the distributions histograms in figure 2.10 reveals the following. Firstly, it is clear that the ionograms are highly quantized. Secondly, this quantization along with dynamic range control has produced a pronounced lack of signal below about 80 in the histogram of the original ionogram resulting in an apparent thresholding of the ionogram. (Note that the threshold of 80 here will not reflect the level at which the ionogram signal was actually thresholded at because I have rescaled the ionogram.) These two limitations on the ionogram data I have would make it difficult to assess what noise model would be most appropriate, but it seems clear that it is dominated by some form of impulsive noise.

It would be advantageous to have ionogram data for processing that has a finer quantization and a larger dynamic range for the following reasons. Firstly, background noise of the type that has been removed by the present analogue to digital conversion process would still not present a problem to any of the noise filtering algorithms I have discussed. Secondly, thresholding is fine for presentation to a human operator, and in some sense I have been forced to do it here (along with rescaling) to assess the performance of the various filters, but it is really premature to do any thresholding and rescaling before the feature detection stage. This would ensure better preservation of information for the feature detection algorithms.

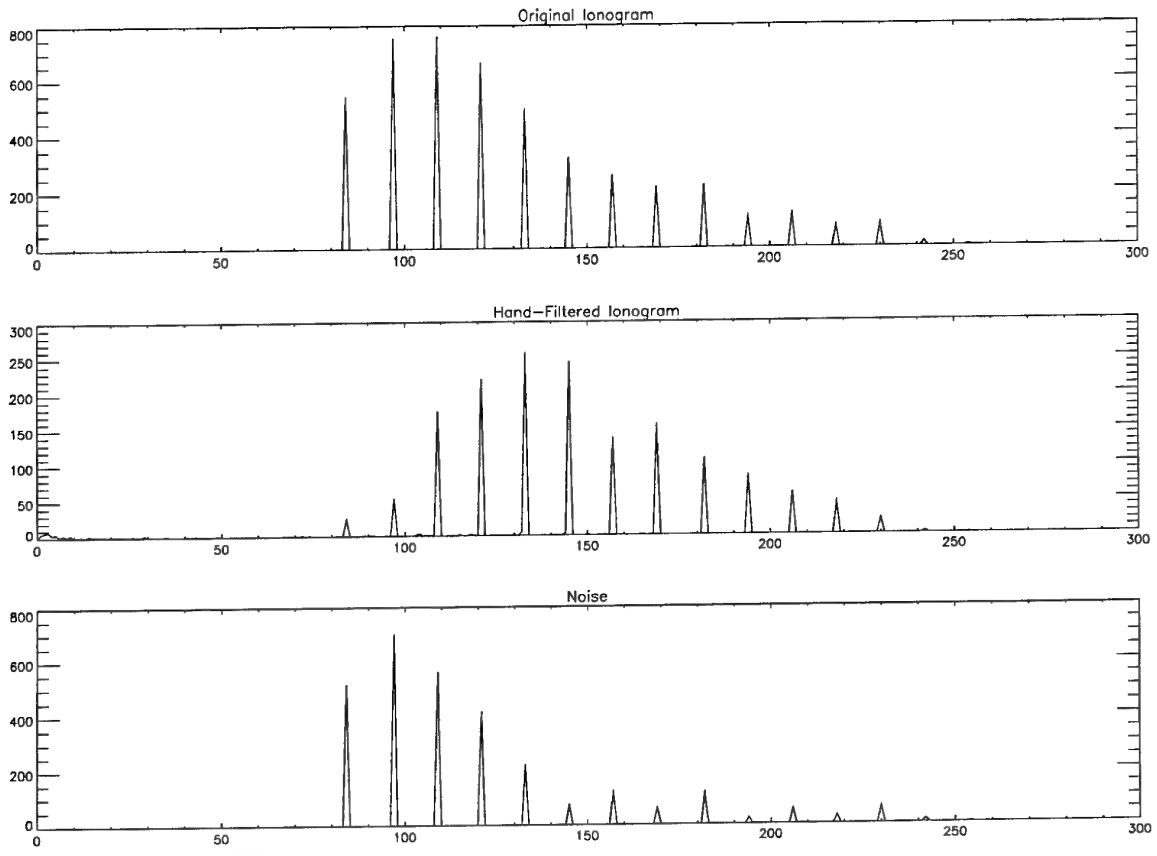


Figure 2.10: The histogram plots of ionogram 1, the hand-filtered version of ionogram 1, and the noise in ionogram 1 based on the hand-filtered signal.

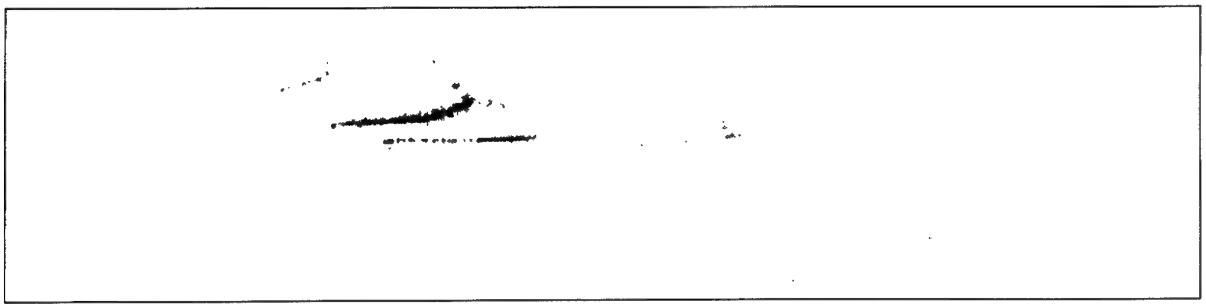


Figure 2.11: Ionogram 1 — filtered by 3 × 3 median filter.

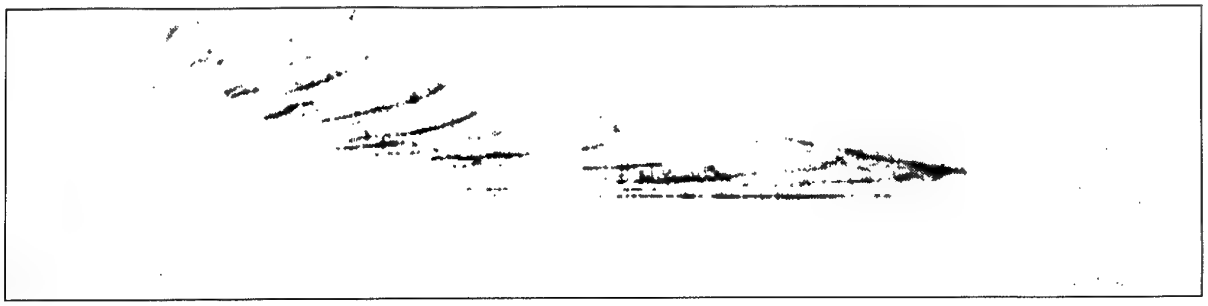


Figure 2.12: Ionogram 2 — filtered by 3 × 3 median filter.

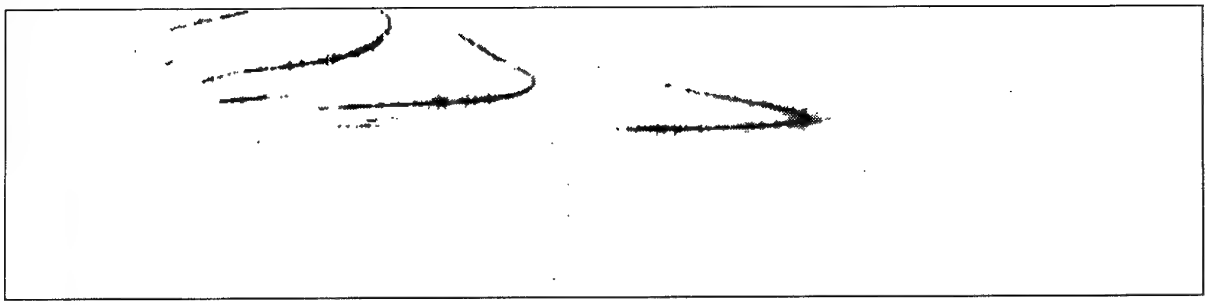


Figure 2.13: Ionogram 3 — filtered by 3 × 3 median filter.

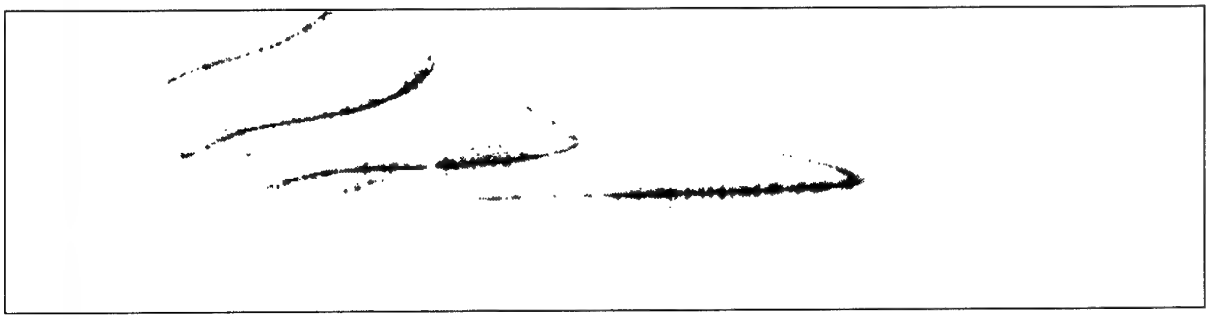


Figure 2.14: Ionogram 4 — filtered by 3 × 3 median filter.

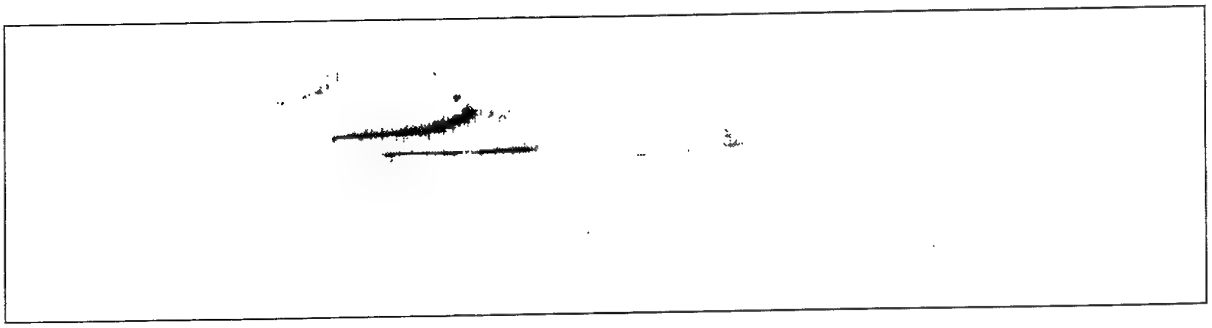


Figure 2.15: Ionogram 1 — filtered by combined median filters.

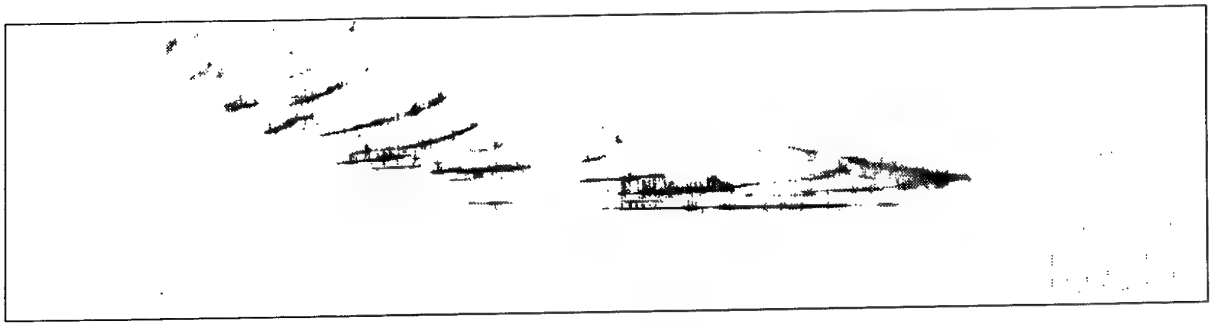


Figure 2.16: Ionogram 2 — filtered by combined median filters.

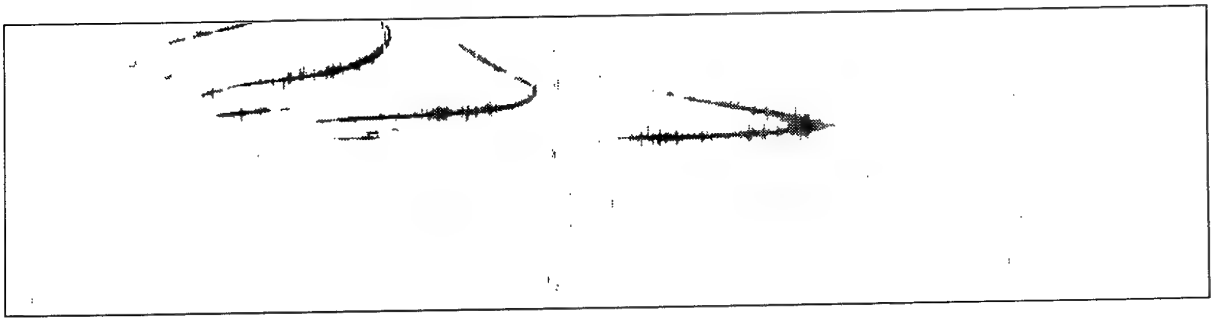


Figure 2.17: Ionogram 3 — filtered by combined median filters.

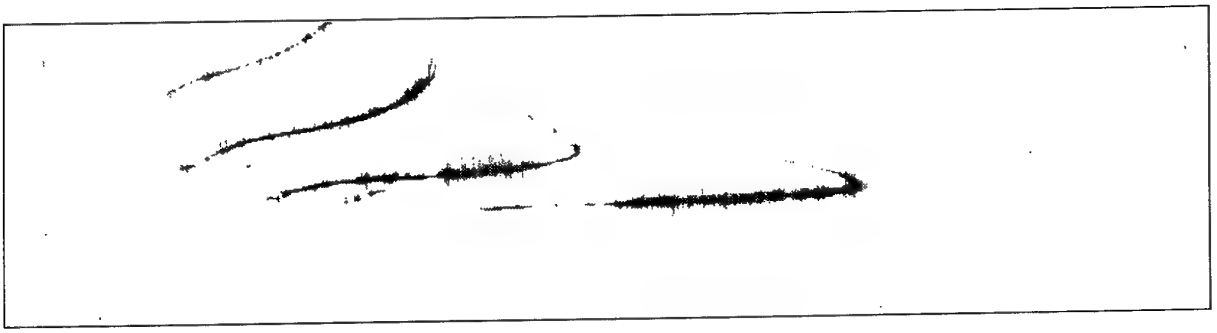


Figure 2.18: Ionogram 4 — filtered by combined median filters.

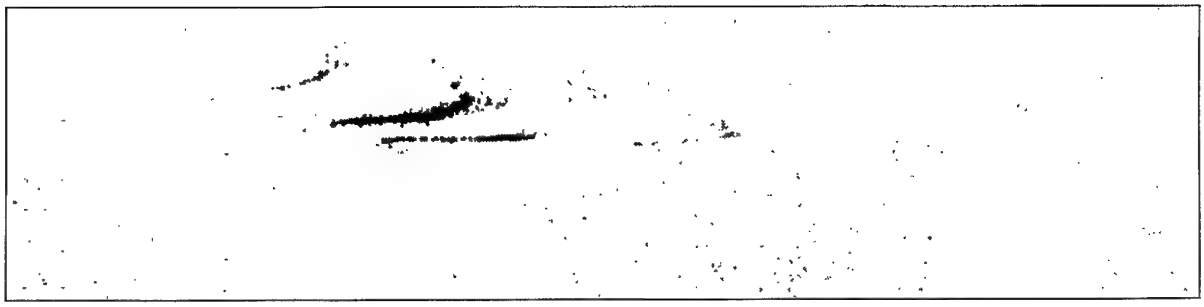


Figure 2.19: Ionogram 1 — filtered by 3×3 quantile filter with $q = 0.75$.

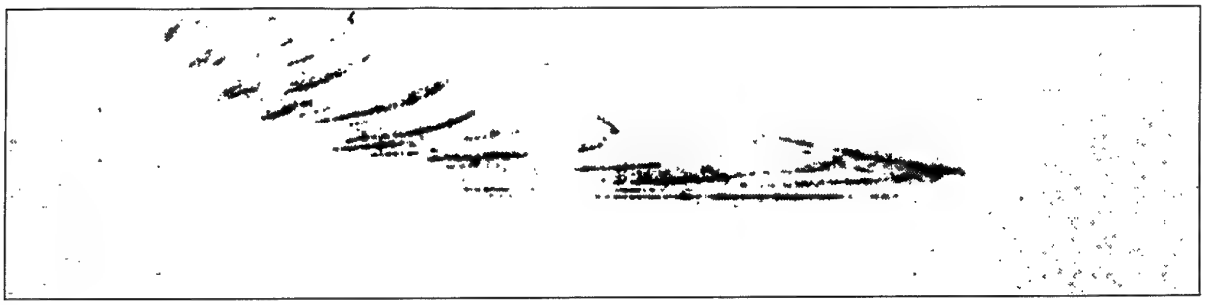


Figure 2.20: Ionogram 2 — filtered by 3×3 quantile filter with $q = 0.75$.

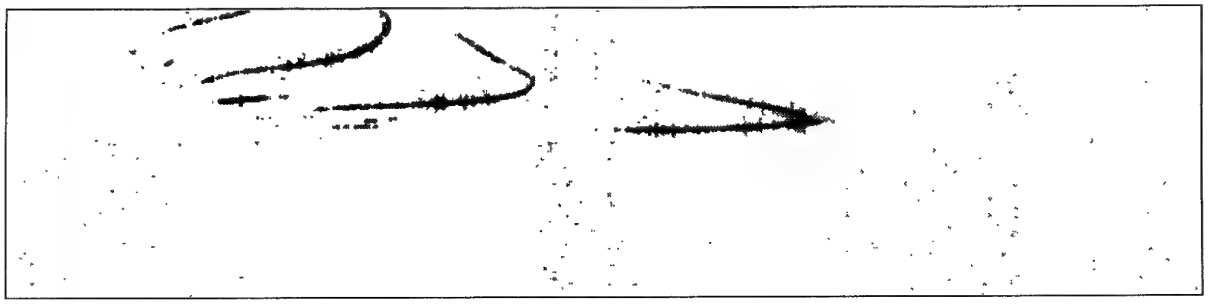


Figure 2.21: Ionogram 3 — filtered by 3×3 quantile filter with $q = 0.75$.

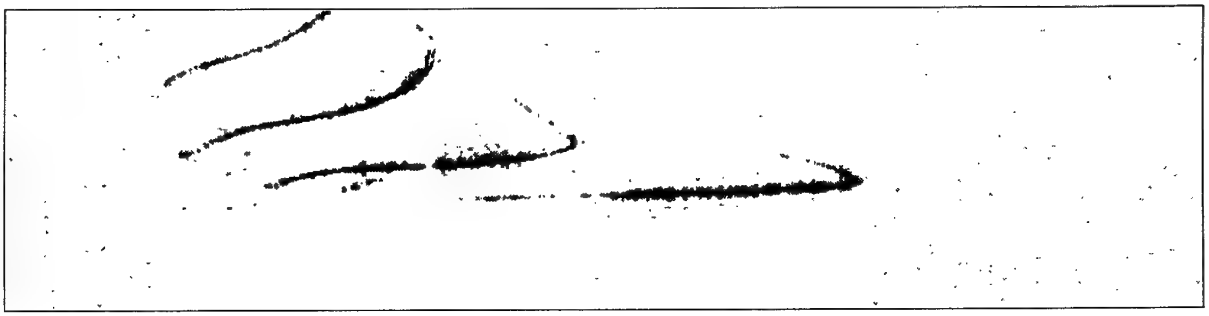


Figure 2.22: Ionogram 4 — filtered by 3×3 quantile filter with $q = 0.75$.

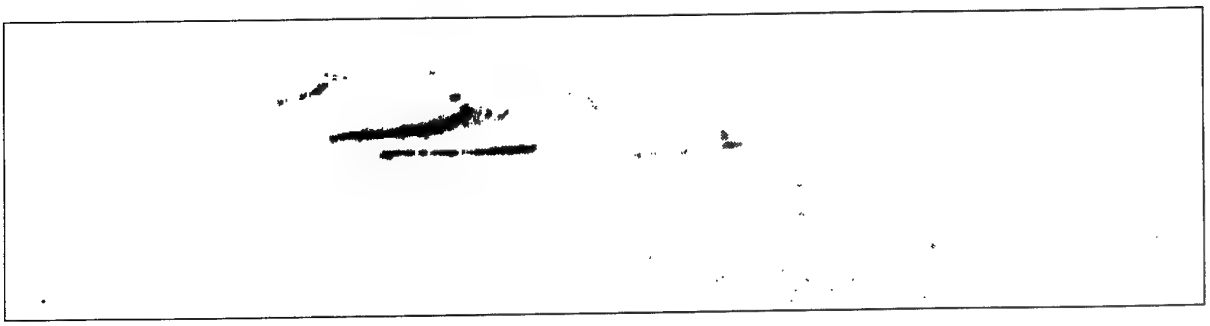


Figure 2.23: Ionogram 1 — filtered by 5×5 quantile filter with $q = 0.8$.

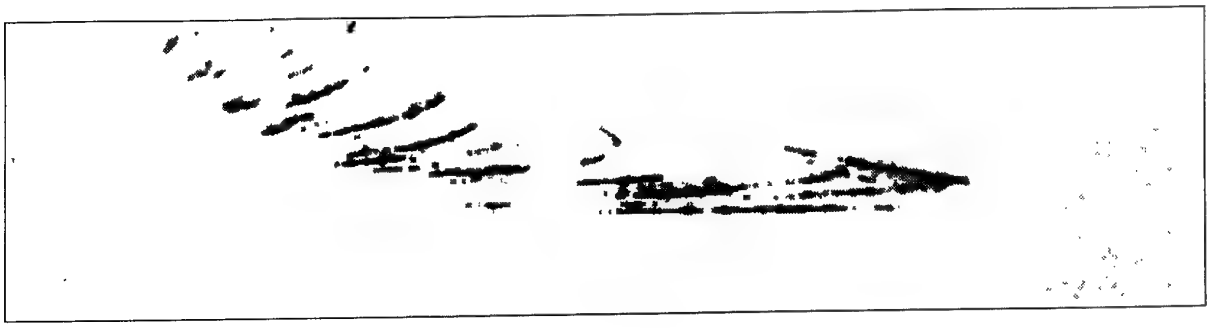


Figure 2.24: Ionogram 2 — filtered by 5×5 quantile filter with $q = 0.8$.

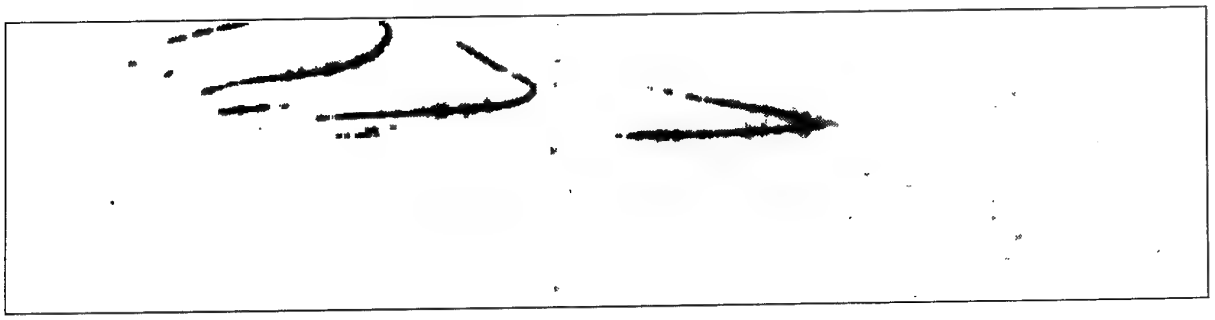


Figure 2.25: Ionogram 3 — filtered by 5×5 quantile filter with $q = 0.8$.

Figure 2.26: Ionogram 4 — filtered by 5×5 quantile filter with $q = 0.8$.

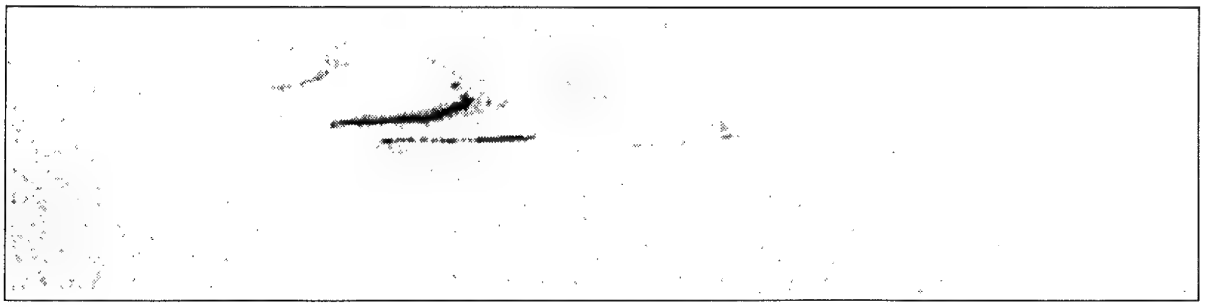


Figure 2.27: Ionogram 1 — filtered by mean and speckle filters.

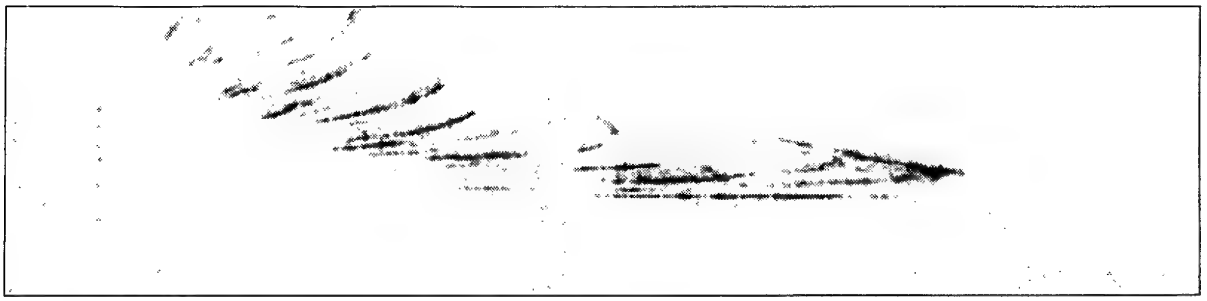


Figure 2.28: Ionogram 2 — filtered by mean and speckle filters.

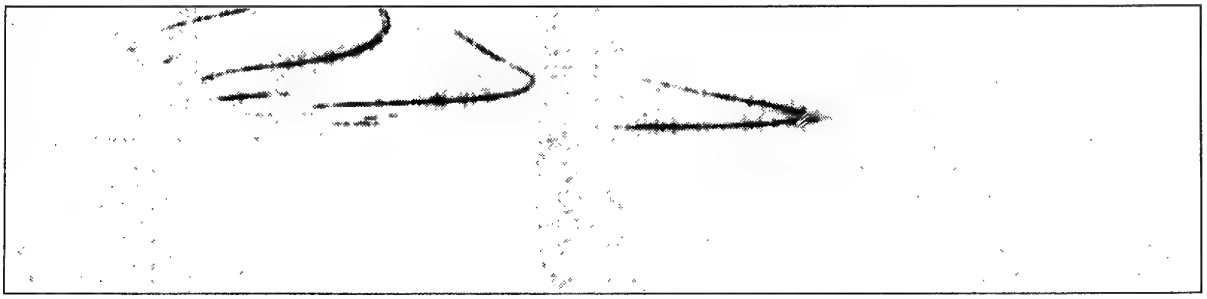


Figure 2.29: Ionogram 3 — filtered by mean and speckle filters.

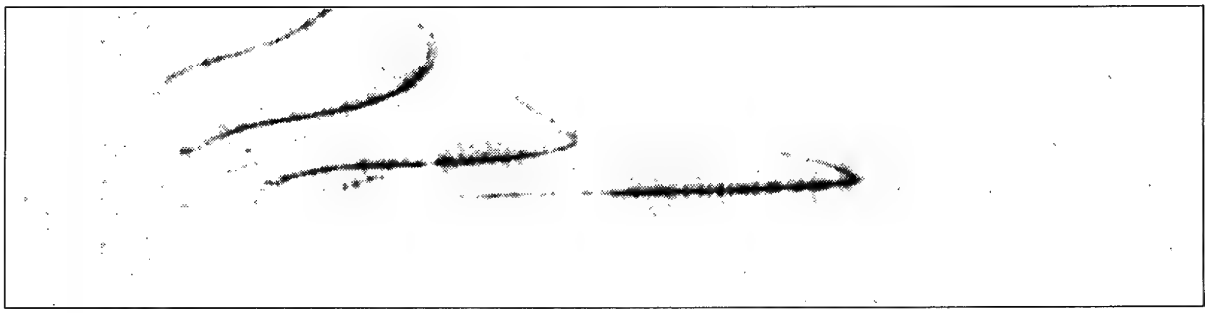


Figure 2.30: Ionogram 4 — filtered by mean and speckle filters.

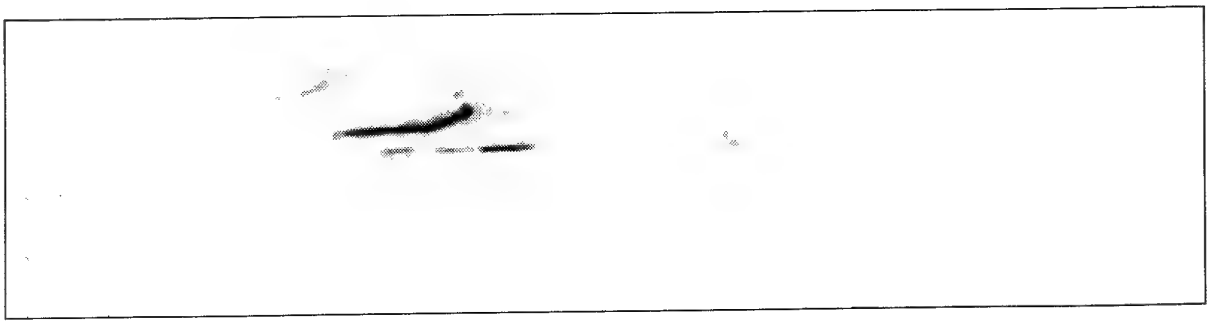


Figure 2.31: Ionogram 1 — filtered by convolution with a circle of diameter 5 and then with a contour smoothing filter.

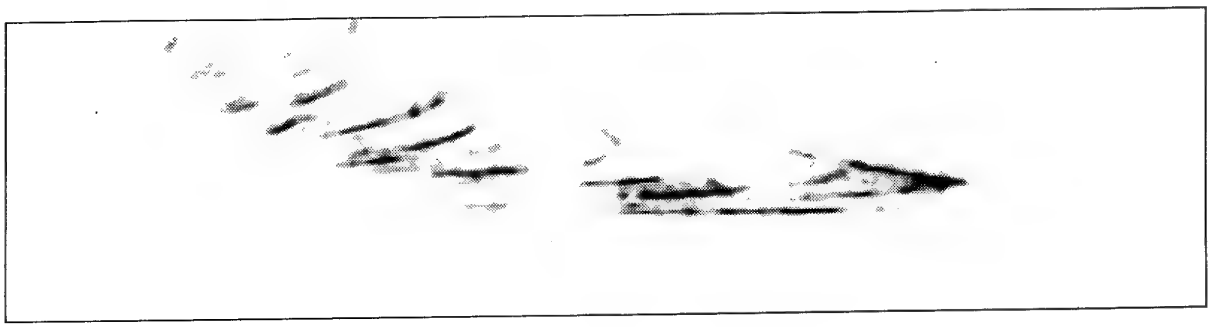


Figure 2.32: Ionogram 2 — filtered by convolution with a circle of diameter 5 and then with a contour smoothing filter.

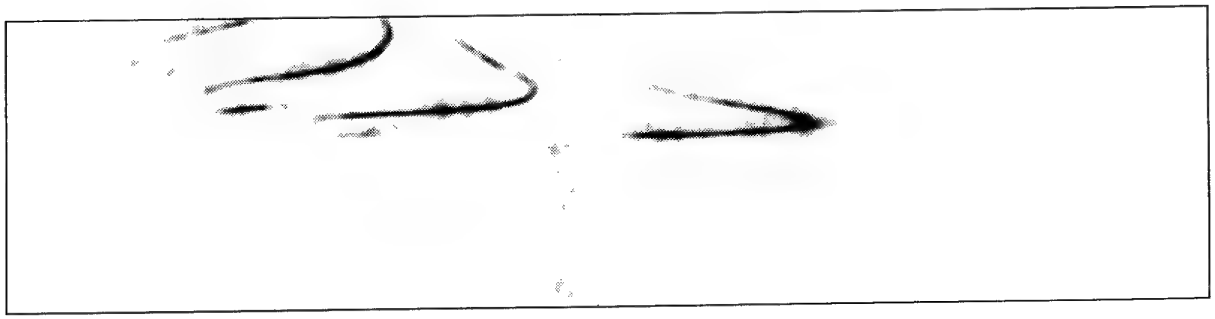


Figure 2.33: Ionogram 3 — filtered by convolution with a circle of diameter 5 and then with a contour smoothing filter.

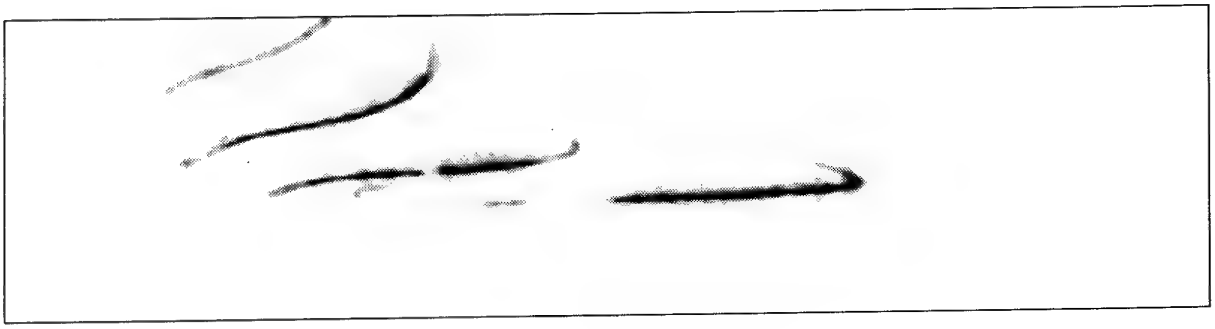


Figure 2.34: Ionogram 4 — filtered by convolution with a circle of diameter 5 and then with a contour smoothing filter.

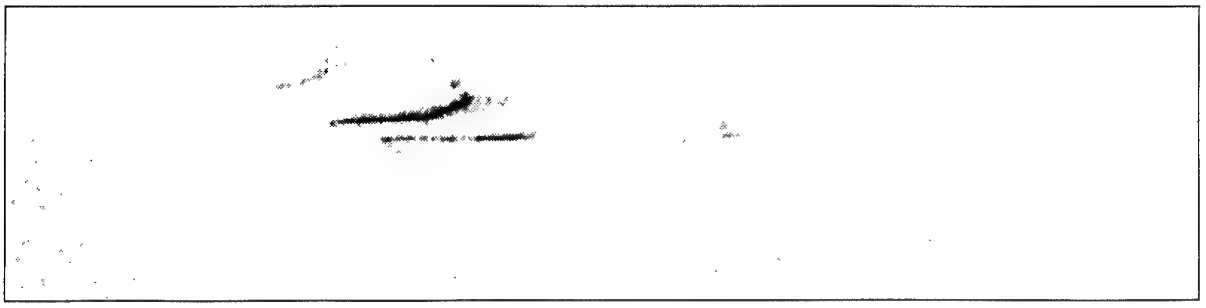


Figure 2.35: Ionogram 1 — filtered by polynomial-fit cleaning filter.

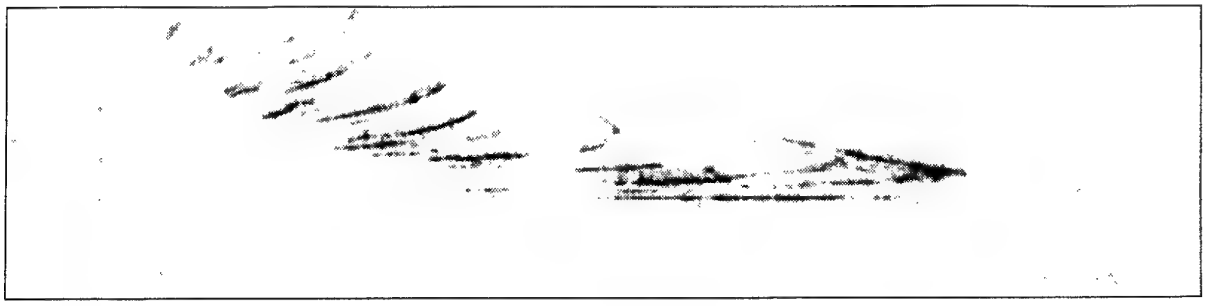


Figure 2.36: Ionogram 2 — filtered by polynomial-fit cleaning filter.

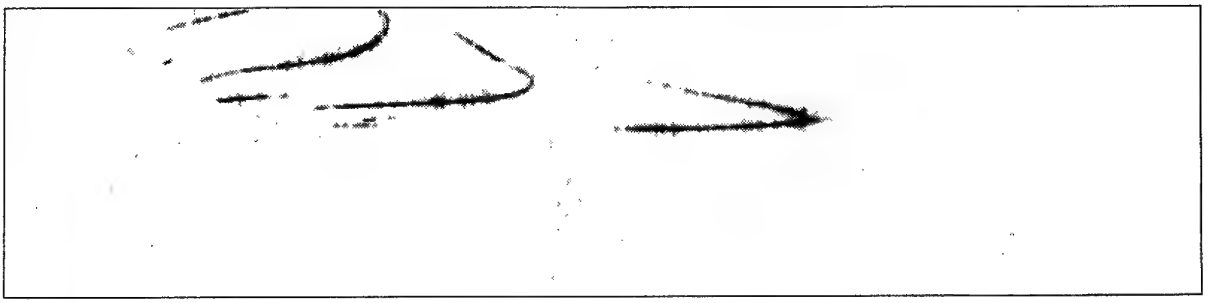


Figure 2.37: Ionogram 3 — filtered by polynomial-fit cleaning filter.

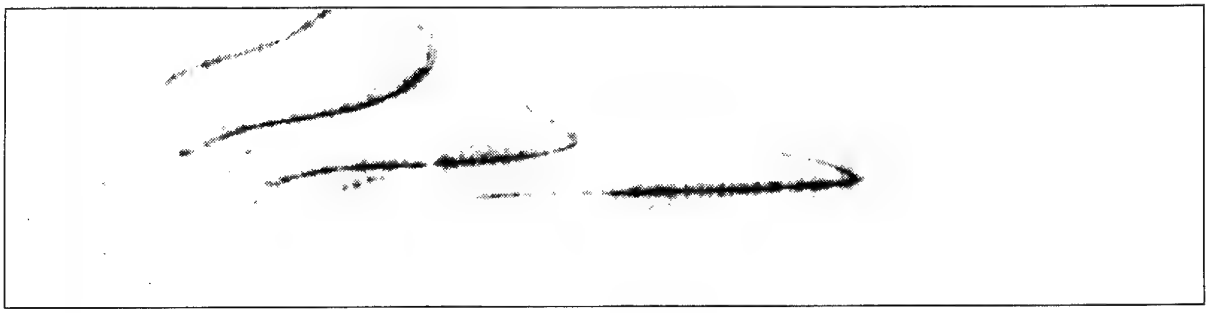


Figure 2.38: Ionogram 4 — filtered by polynomial-fit cleaning filter.

Chapter 3

Feature Extraction: Data Reduction

The most prominent features in an ionogram are the *traces*. These are connected segments in the ionogram and reflect discrete structures and associated modes of signal propagation in the ionosphere: they are clearly visible in figures 2.1–2.4. (Connectivity is defined here to be 8-connectivity, *i.e.*, a pixel may be connected to any of its 8 nearest neighbours including the diagonals). These figures also show, however, that the traces and associated modes can be obscured by noise or other effects, that they can be broken with several distinct traces being obviously associated with the one underlying structure, and that they can take quite complicated forms. Thus the primary aim of the work described in this chapter has been to separate the significant trace shapes so that latter stages can use them to compute feature vectors that summarize trace location and extent for each important propagation mode.

There are two assumptions in dealing with an ionogram as a collection of distinct traces, where traces are defined to be a contiguous area of connected pixels. The first assumption is that a trace will be the product of a single ionospheric propagation mode. Secondly I assume that for each possible propagation mode, there is at most one trace that corresponds to it. Of course, these simplistic assumptions are commonly violated in ionograms, and a number of steps have been incorporated into the autoscaling system to try and cope with these eventualities. The first assumption can be violated when traces are erroneously connected, and the branch selection stage that is discussed in this chapter tries to address this problem. The merging stage, to be discussed in a latter chapter, is designed to deal with those situations where a trace is cut into disconnected segments, breaking the second assumption. In this way we will approach the ideal where all the signal that corresponds to a single propagation mode and no other has been isolated into a single trace of the ionogram. Remember that I will refer to a trace as a trace *structure* when I want to emphasize that the trace contains signal that was received by more than one ionospheric propagation mode, and as a trace *fragment* or trace *segment* when I want to emphasize that it is only a small part of the signal received by a single propagation mode.

The idea behind feature extraction is to find a simple description of the traces in a symbolic form. This is achieved by finding a suitable model such that for some value of the model parameters there is a good correspondence between the shape of the trace and the shape of the model. This is represented in figure 3.1. This highlights a number of problems. Firstly, we need to choose a suitable model for the traces. Secondly, we have to find a measure of the goodness of fit and a fitting routine to determine a value of the model parameter that minimizes the measure. Lastly, we need to separate the pixels that correspond to particular traces and reduce them in number somehow because this will have a serious impact upon the overall speed of the autoscaling system (*i.e.*, the optimization involved in fitting is the rate determining step for speed in the autoscaling system). The last problem, fitting, is our concern in this chapter — the others will be dealt with in successive chapters.

The structure of the complete feature extraction stage is summarized in figure 3.2, which shows the individual steps of the feature extraction process developed for the autoscaling system. The input is a clean ionogram produced by filtering via the polynomial fitting process presented in the previous chapter. The filtered image is firstly treated to a simple step that normalizes the ionogram and then sets all the pixel values below a certain level to zero. This step is performed because of the significant amount of low level noise that still remains in the ionogram after filtering that can interfere and slow down latter stages.

The second step takes the grey-valued filtered image and, in a process called *thinning*, produces a

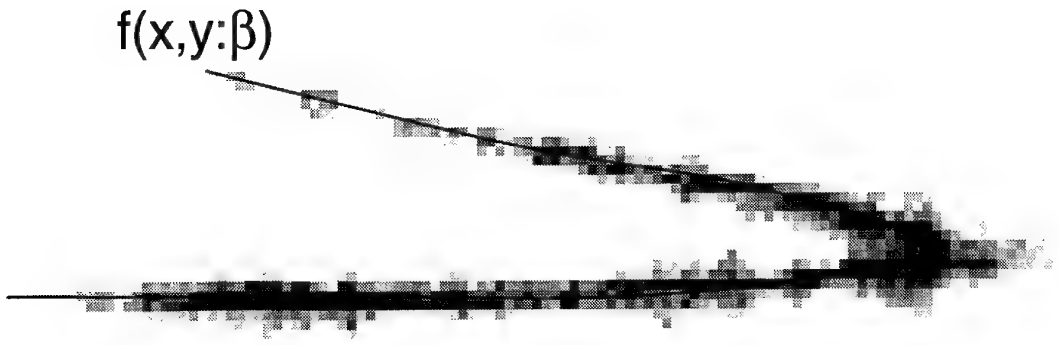


Figure 3.1: The principles of feature extraction. A model f with model parameter β is determined such that for a particular value of β there is a good correspondence between the model and the trace. Feature extraction is the process of choosing a suitable model and determining the best value of β.

skeleton that captures the shape of the traces in a single pixel wide line. The next stage, *trimming*, removes artifacts left by the thinning process. These stages are to reduce the amount of data under consideration.

Lastly, the thinned and trimmed traces are fitted with parameterized curves. The final feature vectors consist of the parameters describing the best fitting curve along with entries giving the start and finish of the trace. The status of an example ionogram at each stage of feature extraction is shown in figure 3.3.

The next sections of this chapter describe in detail the algorithms used to carry out each of these steps, up to and including the trimming stage.

3.1 Test Data

The same four representative LLISP oblique ionograms that were selected for testing filtering algorithms in the previous chapter are used again here: they are shown in figures 2.1–2.4. The traces they contain are a good selection of the types observed in practice. Thus I will use them as a representative set of traces here to report the feature extraction results in a manageable way. (Chapter 7 will present the results of the autoscaling system upon pathological cases.) Nevertheless a caveat should be mentioned. That is, trimming and fitting require certain parameters to be specified. These should be chosen to coincide with natural cut-off points that are apparent in appropriate statistics compiled over all ionograms. As the data set used here is quite small, the values determined from this data set may need to be revised in practice. These parameters, where they exist, will be indicated plainly in the following discussion.

3.2 Zeroing Pixels

Firstly, the dynamic range of the filtered pixel values is stretched so that the maximum pixel value in each ionogram is 255 — this is called *normalization.* This ensures that the threshold used in the next step is always in constant proportion with the maximum signal level. The normalized filtered image is then treated to a simple step that sets all the pixel values below a certain level to zero. The histogram in figure 3.4 depicts the distribution of pixel values for the four filtered test ionograms. The histogram does not show any obvious break or gap indicating a likely separation between noise and signal, but usually those pixels with a magnitude of 20 or less seem to contribute little to the signal of the traces. Indeed,

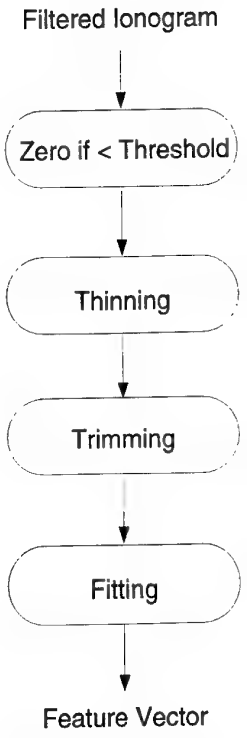


Figure 3.2: Stages used in the feature extraction process.

when these low level pixel values are zeroed out, the result of the thinning stage to follow contains considerably less structures which are due to the presence of noise.

3.3 Thinning

The aim of *thinning* is to reduce each trace to a smaller collection of pixels that still effectively characterize its shape. At the same time the process must preserve the trace's significant features to assist future processing stages. Thinning was chosen since traces are basically line-like objects whose thicknesses are essentially irrelevant to the problem of recognizing it and identifying its associated mode. (Nevertheless, strength and thickness are, of course, important to the problem of determining the bandwidth of any available frequency channel, but we are not concerned at this stage about making these measurements). Therefore thinned traces, which have been transformed from a broad but line-like object many pixels wide to just a single pixel wide skeleton, should retain all the information necessary to recognize modes. At the same time a lot of spurious information should have been removed.

In summary, thinning reduces the amount of data to be considered while preserving a fundamental shape, or *skeleton*, that is topologically similar to the original trace. Thus thinning is also known as *skeletonization*, or alternatively as the *medial axis transform*.

Traditionally, thinning is performed on binary-valued images, and consequently is requires that the ionogram be converted into a binary-valued form. This process will be discussed in a moment. Originally, the thinning stage employed in the autoscaling system was a binary-valued one [9]. However, using an idea by Roughan [10], the binary thinning routine can be simply modified to achieve a good grey-scale thinning algorithm, as we will consider in a moment.

Firstly, though, let us consider binary-thinning, and consequently thresholding which must precede it.

3.3.1 Thresholding

Thresholding is a simple algorithm that converts the filtered ionogram to a binary-valued pixel image.

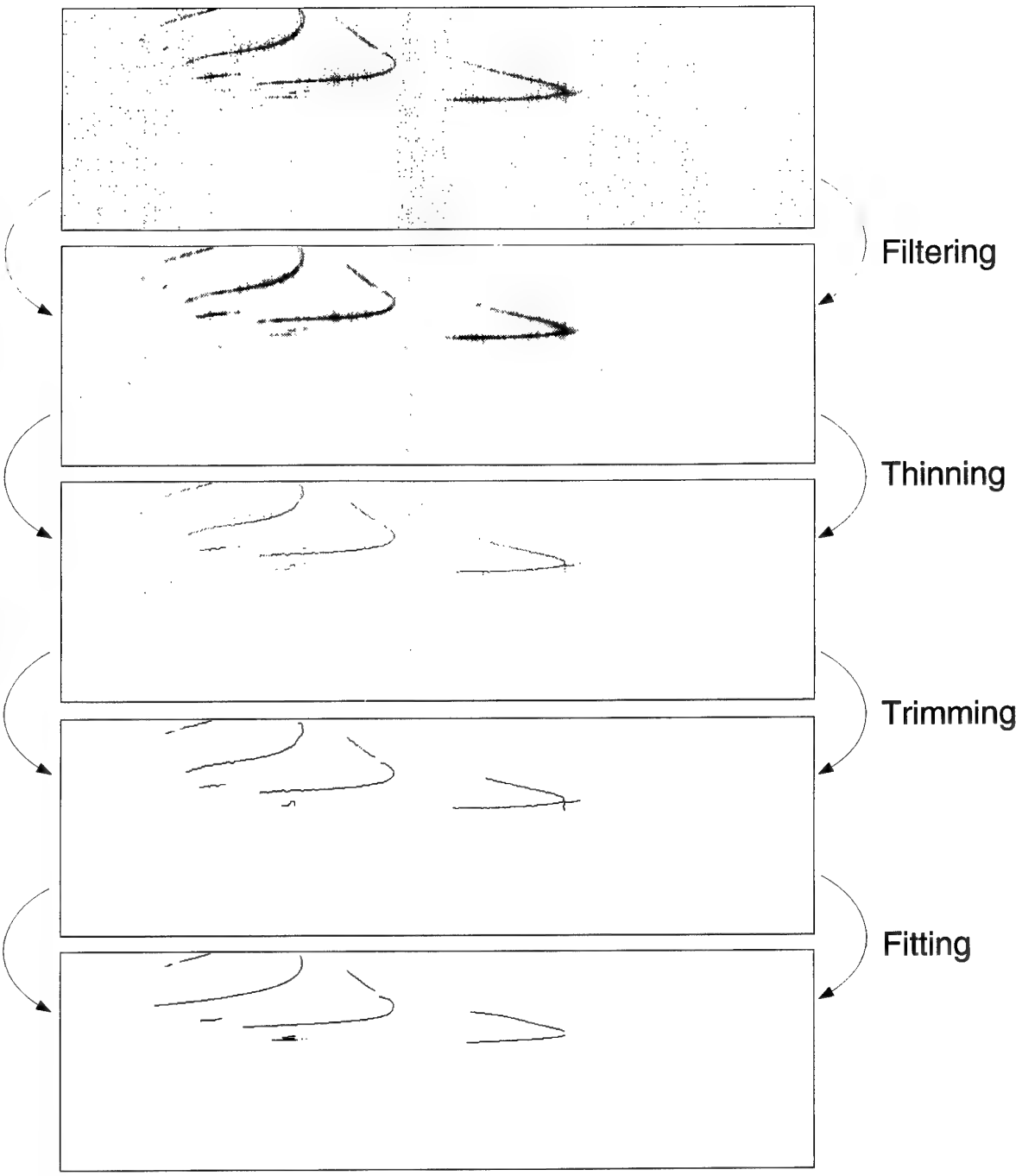


Figure 3.3: One test ionogram at each stage of the feature extraction process.

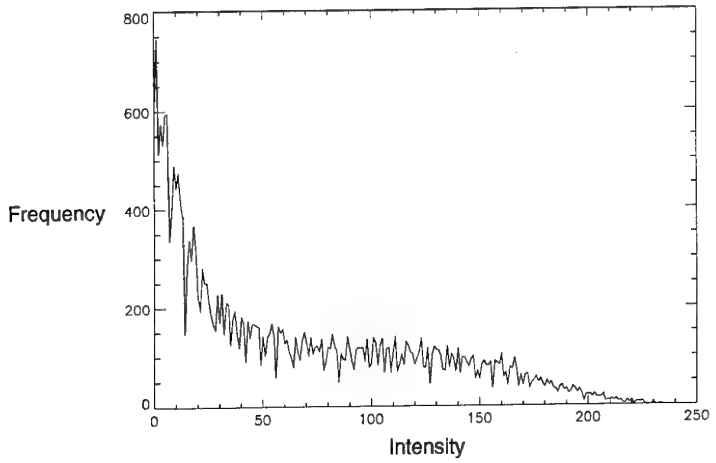


Figure 3.4: Histogram of non-zero pixel values for the four test ionograms after filtering.

All pixel values equal to or above a certain threshold are set to one in the output, and all those below are set to zero. This step is necessary because binary thinning algorithms are designed only to deal with binary-valued data. Because we have zeroed all the pixels below a certain level in the previous step described in Section 3.2, we have in effect decided that every pixel value above this level is relevant for further processing. Therefore, the obvious value for the threshold is a value of one so that every non-zero pixel is retained in the binary-valued pixel image.

3.3.2 Binary Thinning

Binary thinning can be achieved in two general ways [11]. The first approach is based on the fundamental definition of a skeleton as points that are in some sense far from the boundaries. Thus the skeleton of an image can be computed by determining the closest boundary point to every point in an object. If a point is closest to more than one boundary point, then it is included in the skeleton. Figure 3.5 shows the skeleton of a rectangle and how it can be computed by this approach.

This direct approach, however, has two drawbacks. First, it requires each connected component in the image to be identified and processed separately; second, it is computationally intensive. A more efficient approach is to successively delete the edge points of regions until the thinned line is achieved. Edge-point erosion can be effectively carried out by repeatedly traversing the image with a 3×3 window. Erosion in each window is performed according to a set of rules to ensure three things: first, end points must not be removed; second, connectedness must be preserved; and third, excessive erosion of regions must not occur [11]. As the approach uses only local operations, it is computationally efficient and does not require prior segmentation of the image into separate regions.

The algorithm used here is due to Zhang and Suen [11,12], and involves successive passes of two basic edge erosion operations. It assumes that the image is binary-valued, and proceeds by labelling pixels in the 3×3 window according to the following regime:

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

Two measures of the neighbourhood of the pixel p_1 are used: the first, denoted by $N(p_1)$, indicates the number of nonzero pixels $p_2, p_3, \ldots, p_9$; the second, $S(p_1)$, indicates the number of 0–1 transitions in the ordered sequence $p_2, p_3, \ldots, p_9$.

The first erosion operator flags a pixel for deletion if the following conditions are met:

1. $2 \leqslant N(p_1) \leqslant 6$
2. $S(p_1) = 1$
3. $p_2 \cdot p_4 \cdot p_6 = 0$

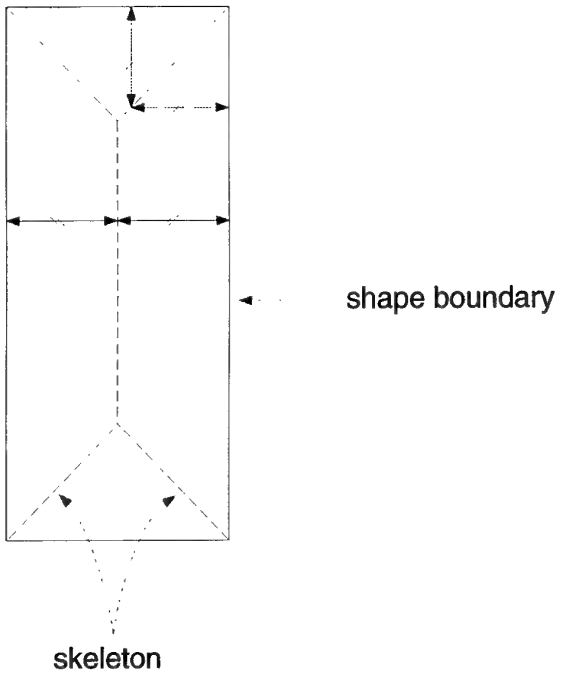


Figure 3.5: Example of the points that form a skeleton.

4. $p_4 \cdot p_6 \cdot p_8 = 0$

Flagged pixels are deleted only at the end of a complete pass through the entire image by the operator. The first condition ensures that an end-point of a thinned line is not deleted (corresponding to the case when $N(p_1) = 1$) and that regions are not eroded (corresponding to the case when $N(p_1) \geqslant 7$). The second condition is violated when the window contains a line that is only one pixel wide, and so ensures that a thinned line is not broken by the thinning procedure. The conditions $p_2 \cdot p_4 \cdot p_6 = 0$ and $p_4 \cdot p_6 \cdot p_8 = 0$ will be satisfied whenever the pixels p_2, p_4, p_6, p_8 are in any configuration *except* the following

	1	
1		0
	1	

	1	
1		1
	0	

	1	
1		1
	1	

and so any pixel p_1 with such neighbours will not be deleted.

The second pass is very similar to the first and marks pixels for deletion that satisfy the following:

1. $2 \leqslant N(p_1) \leqslant 6$
2. $S(p_1) = 1$
3. $p_2 \cdot p_4 \cdot p_8 = 0$
4. $p_2 \cdot p_6 \cdot p_8 = 0$

Once again, flagged pixels are deleted only at the end of a complete pass through the entire image. The conditions $p_2 \cdot p_4 \cdot p_8 = 0$ and $p_2 \cdot p_6 \cdot p_8 = 0$ will be satisfied whenever the pixels p_2, p_4, p_6, p_8 are in any configuration *except* the following

	0	
1		1
	1	

	1	
0		1
	1	

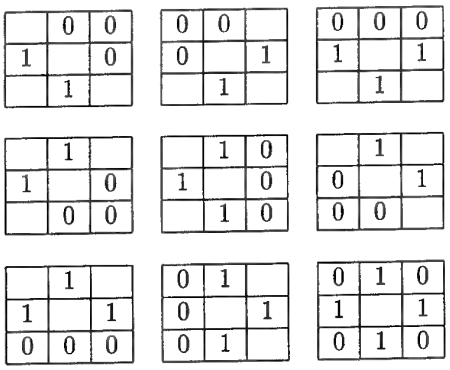
	1	
1		1
	1	

and so any pixel p_1 with such neighbours will not be deleted.

The first and second passes are repeated alternatively until no more pixels can be removed. Once this stage is reached, one last cleanup removes any unnecessary corner pixels from the thinned lines [13]. This is carried out in a single pass across the image by a third 3×3 window that looks for pixels satisfying the following condition

$$p_2 \cdot p_4 \cdot \bar{p_7} \vee p_4 \cdot p_6 \cdot \bar{p_9} \vee p_6 \cdot p_8 \cdot \bar{p_3} \vee p_2 \cdot p_8 \cdot \bar{p_5} = 1.$$

This condition ensures that all pixels p_1 with neighbours in any of the following configurations



will be deleted.

For more details on the complete thinning algorithm, see the pseudo-code below in algorithm 1.

Algorithm 1

```
01    do
02        % pass one
03        for each row
04            for each column
05                place window
06                if p1 = 0
07                    continue
08                end if
09                if 2 ⩽ N(p1) ⩽ 6 and S(p1) = 1
10                    if p̄4 or p̄6 or p̄2 · p̄8
11                        mark for deletion
12                    end if
13                end if
14            end for
15        end for
16        delete marked pixels
17        % pass two
18        for each row
19            for each column
20                place window
21                if p1 = 0
22                    continue
23                end if
24                if 2 ⩽ N(p1) ⩽ 6 and S(p1) = 1
25                    if p̄2 or p̄8 or p̄4 · p̄6
26                        mark for deletion
27                    end if
28                end if
29            end for
30        end for
31        delete marked pixels
```

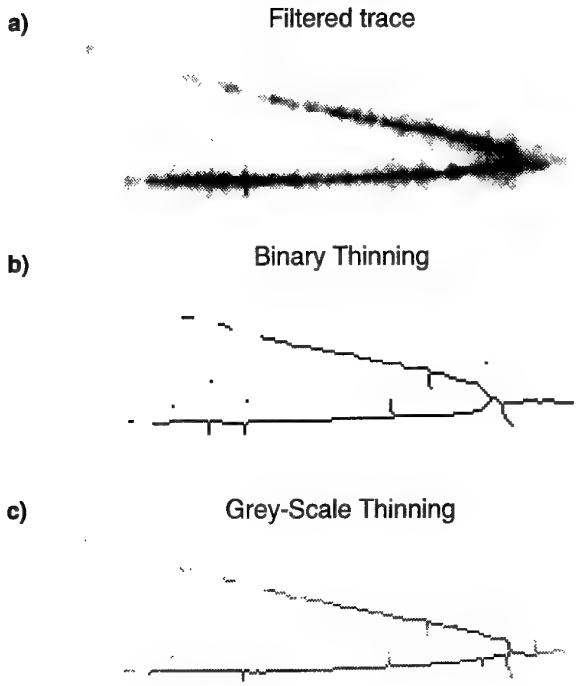


Figure 3.6: A comparison of the binary thinning routine I originally proposed to use for the autoscaling system against the new grey-scale approach. Note that the trace and its two skeletons are lined up vertically, and that the binary thinning routine has not produced a skeleton that passes through the high intensity region of the trace around the nose.

```
32      until no deletions
33      % remove corner points
34      for each row
35          for each column
36              if p2 · p4 · p̄7 or p4 · p6 · p̄9 or p6 · p8 · p̄3 or p2 · p8 · p̄5
37                  delete pixel
38              end if
39          end for
40      end for ■
```

3.3.3 Grey-Scale Thinning

The binary thinning approach has problems in its treatment of the traces in an ionogram, and this can be seen in figure 3.6. In this example, which depicts an enlargement of a trace around the junction frequency, we can see how the binary thinning approach has failed to correctly capture the true shape of the trace because it has failed to track the trace around the nose along the high intensity ridge-top. This is because in binary thinning all information about the intensity of any pixel is not utilized, and consequently we need to consider a *grey-scale* thinning algorithm.

A method to generalize binary-thinning algorithms to give grey-scale thinning is presented in [10]. This is simply achieved in the following way. Firstly, a set of thresholds $\{t_i : i = 1, \ldots, N\}$ are chosen such that $t_i < t_{i+1}, i = 1, \ldots, N-1$ and $t_N = 255$, where 255 is the largest pixel intensity in the image because of the normalization already performed on the ionogram. Next, a series of binary thinnings are performed according to algorithm 1 above (or indeed, any other binary thinning routine will also serve) with the additional condition that for the i-th application of the routine a pixel is only removed if its intensity lies in $[0, t_i]$. That is, on the first application, in the edge erosion operators in the first and second pass only flag a pixel p_1 for removal if its intensity $I(p_1)$ satisfies $I(p_1) \in [0, t_1]$ in addition

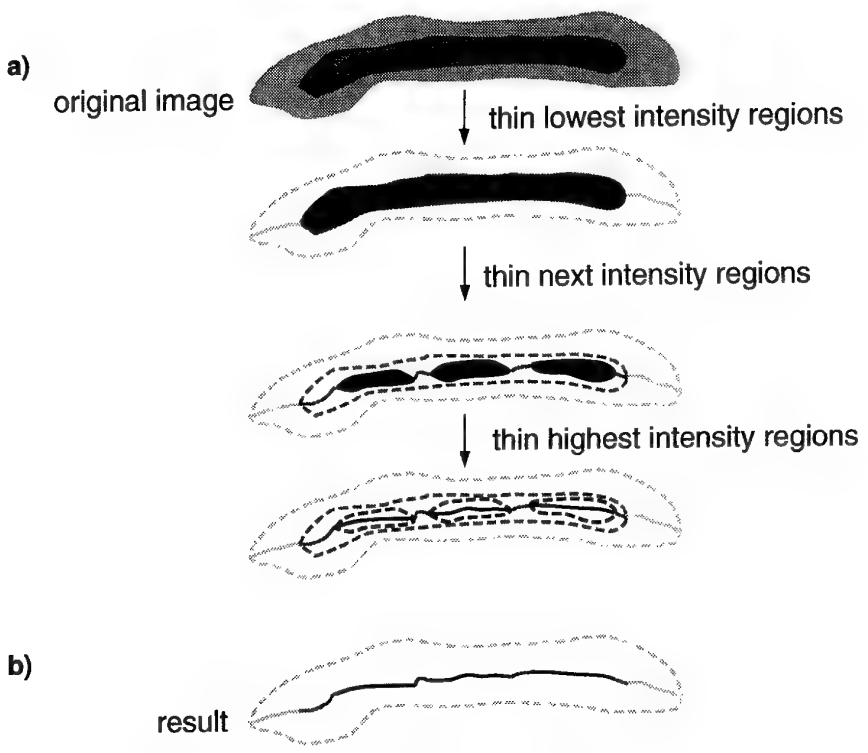


Figure 3.7: Constructing a grey-scale thinned object using a binary thinning algorithm.

to the four conditions already presented for each pass, and so on until the modified binary thinning routine has been applied N times and no more pixels can be removed. This process has been represented diagrammatically in figure 3.7.

The only matter that needs to be settled now is the manner in which we select the thresholds $\{t_i : i = 1, \ldots, N\}$, and their number N. This has been achieved using an automatic threshold selection algorithm due to Whatmough [14]. Essentially, this algorithm firstly places a convex hull over the log of the histogram of pixel intensities in the ionogram. Then, the algorithm seeks to minimize the area (actually, the exponential of the area) between the convex hull and the histogram in the following way. Think of the convex hull of the log histogram as a rubber band stretched tight across the log histogram and tied to the x-axis at the origin and 255, the maximum pixel intensity (see figure 3.8(a)). Then, imagine that a number of weights (up to the maximum N) are suspended on strings from the rubber band at particular intensities so that the rubber band is weighted down until it hits the log histogram at that particular intensity without moving laterally (see figure 3.8(b)). This will have produced a reduction in the exponential area between the rubber band and the log histogram. The final threshold values are determined by a dynamic programming procedure that adjusts the position in intensity and number (up to N) of the thresholds until a minimum exponential area is achieved between the rubber band and the log histogram.

The next matter to consider in the automatic threshold selection is how many thresholds N are necessary. Figure 3.9 shows the results of using 6 versus 16. Note that in the case of 16 levels, the grey-scale thinning procedure has done a slightly better job of following the high intensity regions of the separated x and o-ray. Qualitatively, it was found that 16 levels produced a noticeable improvement over a smaller number of levels, whilst more than this number did not seem to have appreciable effect. This is not surprising when one considers the quantized nature of the pixel intensity of LLISP ionograms (see figure 2.10). Therefore a maximum of 16 levels was settled upon for the grey-scale thinning of LLISP ionograms using automatic threshold selection.

The four grey-scale thinned ionograms produced by applying this algorithm are shown in figures 3.22–3.25.

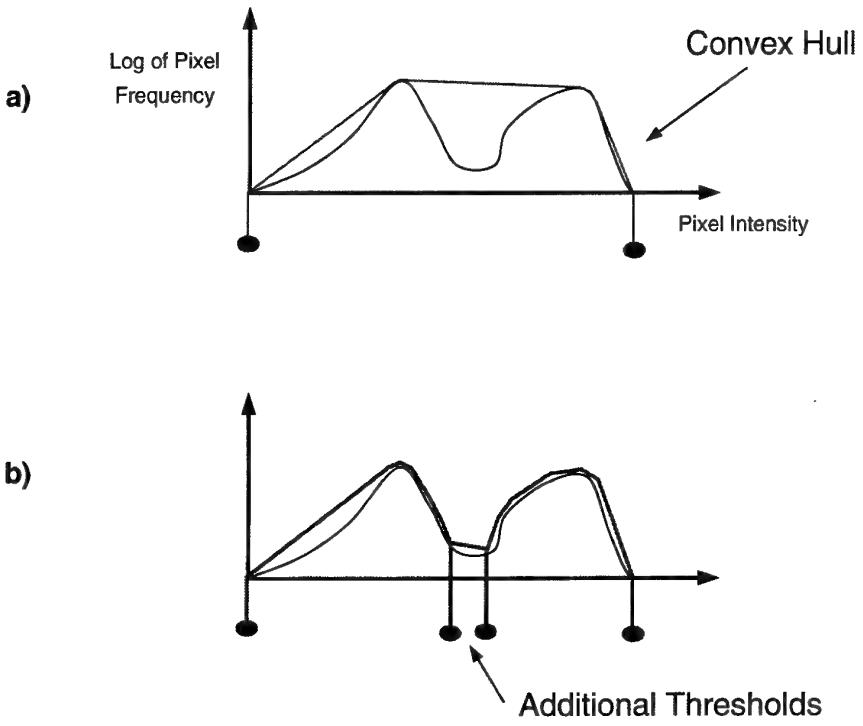


Figure 3.8: Automatic threshold selection via the exponential hull method.

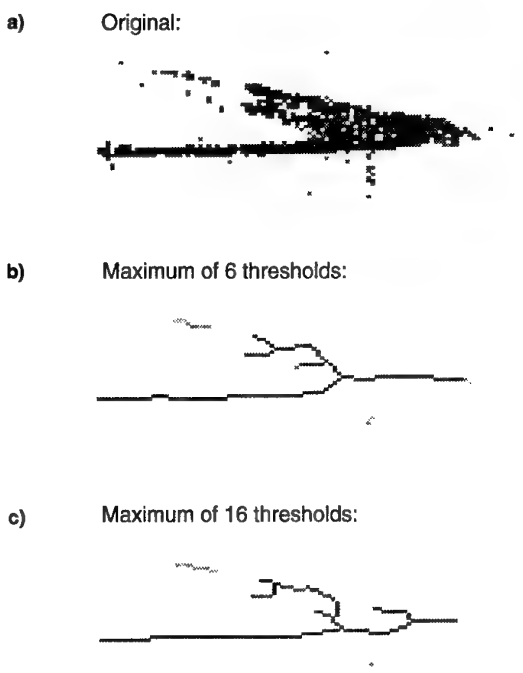


Figure 3.9: A comparison of the effects of using different numbers of levels in the grey-scale thinning algorithm.

A Faster Grey-Scale Thinning

The time required to produce a grey-scale skeleton by the approach presented above could become significant, particularly in the case of 16 levels as is used here, if one needs to obtain the fastest possible autoscaling system. Therefore the following ideas, which are an original extension of Roughan's scheme for generalizing binary thinning algorithms to produce grey-scale skeletons [10] would be worthwhile implementing. They are similar in some regards to the grey-scale thinning algorithm of [15].

Consider what would happen if the number of levels in the grey-scale thinning algorithm above was increased until it equalled the number of grey levels in the image, which in our case is 255. If this were to occur, the values of the set of thresholds $\{t_i : i = 1, \ldots, N\}$, where $N = 255$, would be obvious and natural, so that no information in the grey levels of the image was lost. However, the cost of repeatedly applying what is effectively a binary thinning algorithm would be prohibitive.

We can produce the same result by a much speedier approach in the following way. In the grey-scale algorithm above we perform a number of loops across the entire image (the same in number as a single application of a binary thinning algorithm) N times. Instead, we can consider the image as a list of pixels in ascending order of their pixel intensities. All we have to do is look at the beginning of our list for pixels p such that $I(p) \in [0, t_1]$ which might satisfy the erosion conditions instead of searching through the entire image. All such pixels are considered before removal in the usual manner using two passes until no more can be marked for deletion as before. Then the pixels of the next highest unit of intensity are considered on their own: if none are marked for deletion, then move on to the next highest intensity. If some have been marked for deletion, then re-consider all the pixels of lower intensity, because a higher pixel's removal may free up a lower one for deletion. This process is repeated until pixels of the highest intensity have been considered. These tests could be made very fast by various book-keeping data structures. In this way, the repeated looping through the ionogram is avoided using an ordered list of pixel intensities that combines a pointer to the pixel's location in the ionogram.

3.4 Trimming

The thinned ionograms in figures 3.22–3.25 still contain two particular types of unwanted rubbish. First, there are a number of small isolated lines that are due to noise that were not removed by filtering. Second, a lot of small "spines" occur along the major thinned lines. Obviously, the removal of these artifacts would greatly improve the quality of the thinned ionograms. This can be done by the rather straight forward process of *trimming*.

Trimming starts from the assumption that the longer a thinned line is, the more likely it is to be significant (*i.e.*, a portion of a thinned trace) rather than inconsequential (a spine or thinned noise). Therefore, if lines or spines are shorter than a certain critical threshold, they should be removed.

Some definitions are needed to express this process precisely. A thinned ionogram is said to be composed of *branches*, which are caused by noise, spines, or the thinning of true traces. The end-points of a branch are called *vertices* and are at the end of a line or a junction of two or more branches. They are pixels that have three or more neighbours or only one neighbouring pixel. The *major branches* are those branches that contribute to the representation of the essential trend or shape of the ionogram trace — that portion that a human can clearly detect is relevant at a glance. All other branches are termed *minor* branches, some of which are *spines* and are an artifact of thinning, or *spurious* because they do not contribute to the representation of the important details of the trace's shape. Further, any branch that has only one common end-point with another branch is termed a *terminal branch*. Those end-points that lie on the end of a terminal branch that have only one neighbour are called *terminal points*. These concepts are illustrated in figure 3.10.

The aim, then, of trimming is to distinguish between major and minor branches of a single large trace structure and remove all minor branches from it. In addition, we also want to remove the fragmentary thinned structures in the thinned ionogram that are due to the presence of noise[1]. If these are small in total length, then they should be completely removed. An obvious approach to achieving these two goals is to remove all branches from a trace structure that have length less than a critical threshold, and all contiguous major branches which have accumulated length less than a critical threshold. It remains to choose a suitable value for this threshold. If the lengths were distributed according to a mixture of two

[1] Sometimes these may not be due to noise but are too small to be distinguished from it.

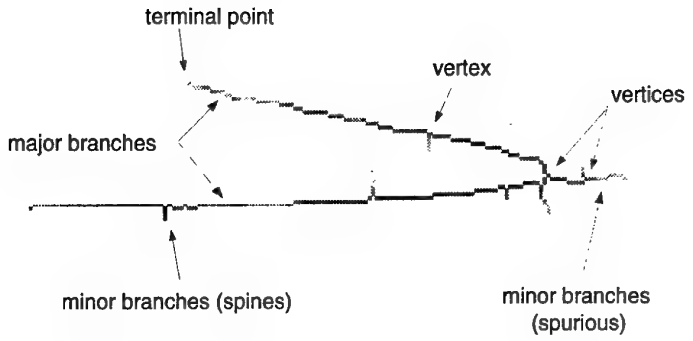


Figure 3.10: Examples of structures in a thinned trace.

distributions, one for the minor branches and one for the major branches, then it is reasonable to hope that the distribution of branch lengths would be bimodal, and the location of the minimum between the two peaks would be a good value for the critical threshold. Figure 3.11 depicts the histogram of branch lengths for our four test ionograms.

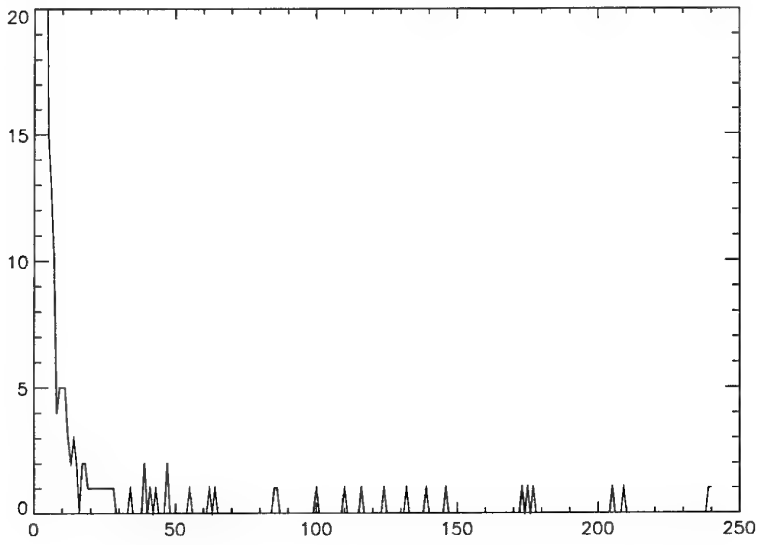


Figure 3.11: A histogram of branch lengths of the four thinned test ionograms. The histogram values for branch lengths in the range 0–4 are off scale.

Unfortunately the actual distribution of lengths appears to be monotone decreasing (and might be most appropriately modelled by an exponential distribution): it is not bimodal so there is no clear choice of threshold. Therefore the choice will be somewhat arbitrary and must be made on the conservative side to ensure that no short lines corresponding to trace fragments are removed. In the algorithm here the threshold was set to remove all minor branches that have a length of 10 or less pixels. The results of trimming with this value using the simple minded approach presented above can be seen in figures 3.26–3.29.

On badly behaved ionograms this trimming approach can fail, and this occurs when a spurious branch near the end of a trace's skeleton is actually longer than the branch that really corresponds to the end of a trace. When this happens, the shorter true component is deleted in preference to the longer spurious component. This is demonstrated in figure 3.12. Situations like this can also happen because loops are not explicitly dealt with in the implementation, and are broken arbitrarily at one of the junctions, so that when trimming occurs, a broken loop can form the longest pathway, and the actual true ends of the

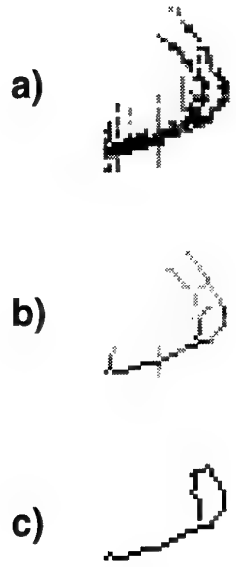


Figure 3.12: The presence of a loop and magneto-ionic splitting has caused the trimming algorithm to fail in this case because spurious branches near the end of the trace skeleton form a pathway that is actually longer than the true trace branches near the end points.

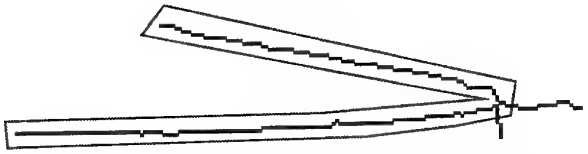


Figure 3.13: Branch selection by choosing those branches of the thinned-trimmed ionogram trace that correspond to the longest pathway through the structure.

traces are cut off. Therefore this problem could be partially addressed by detecting loops in the trace structure and dealing with them appropriately. However, I will introduce shortly a far more sophisticated approach to trimming.

3.5 Branch Selection

The result of thinning and trimming the traces of a filtered ionogram by the processes we have just considered is a branch structure that can still be rather complicated. It will contain a principal set of branches that correspond to the important shape of the ionogram trace, plus other branches that are not relevant to the shape but are longer than the length threshold used in the preceding trimming stage. *Branch selection* is the process of choosing the branches of the thinned-trimmed traces that are deemed to be important to the shape of the trace. These components will be used in the next stage of feature extraction described in the following chapter. Currently, branch selection is performed using the simplistic approach of choosing those branches in the structure that correspond to the longest pathway. This approach is depicted in figure 3.13.

Sometimes these branches can be difficult to interpret due to the high adjacency of the traces, which results in different traces that are due to different propagation modes being erroneously joined in the branch structure of the thinned-trimmed traces. A bad example of this behaviour can be seen in figure 3.14. Furthermore, sometimes noisy and diffuse areas of the trace can produce long spurious branches. (See figure 3.15 for an example of these spurious branches.) Cases like these will present

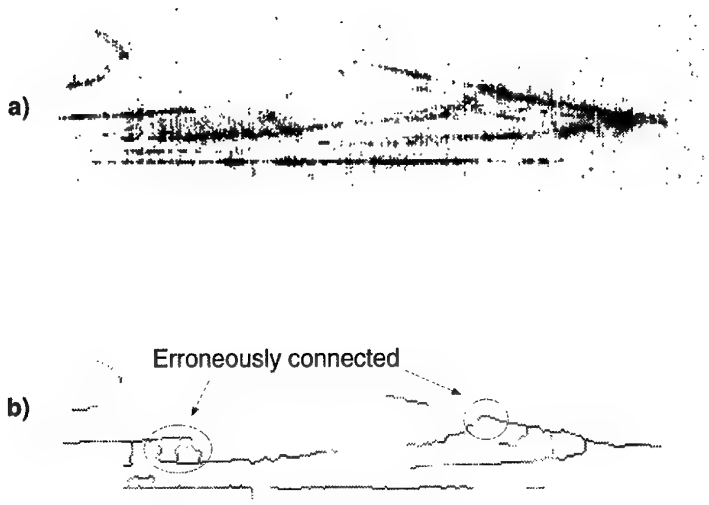


Figure 3.14: Erroneously connected and difficult to interpret branch structures due to traces of high adjacency. This pathological case is difficult to interpret even for a human expert. a) region of unfiltered ionogram. b) Trimmed, thinned, filtered version of a).

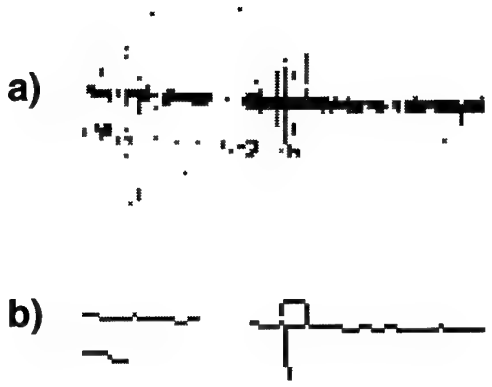


Figure 3.15: An example of spurious branches in the branch structure of thinned-trimmed ionogram traces. a) An enlargement of region of an unfiltered oblique ionogram. b) An enlargement of the same region of the thinned and trimmed version of the ionogram.

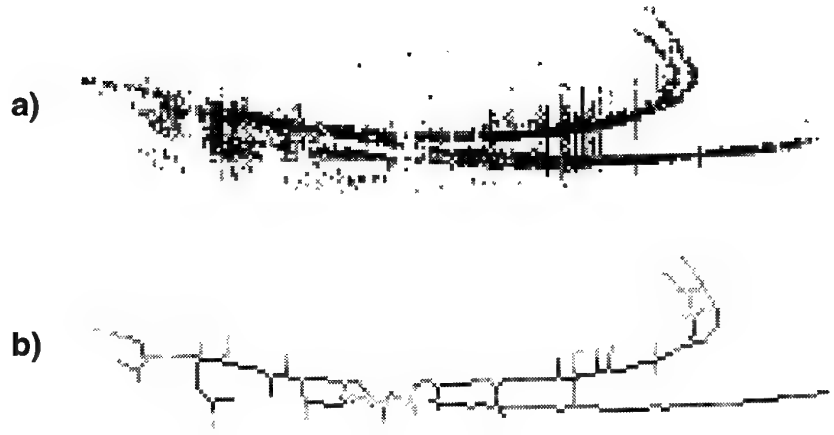


Figure 3.16: A thinned ionogram trace structure that contains components from more than one propagation mode.

problems for the simple-minded approach to branch selection proposed above.

In Chapter 7, there is a detailed assessment of the performance of the autoscaling system on a collection of "interesting" (*i.e.*, difficult) ionograms, which were independently chosen by an expert user. These ionograms highlight the need to implement a more sophisticated approach to branch selection.

I will outline shortly how a more sophisticated branch selection could be implemented.

3.6 Improved Trimming

The approach I propose for improved trimming is based on the observation that the trimming of branches should only take place if they are "perpendicular" to the direction of the trace at that point. This could be implemented in the following way, but time constraints have not allowed me to include this in the system developed so far [2].

Consider the difficult case of the thinned ionogram trace in figure 3.16. This thinned structure is clearly a difficult example for the autoscaling system to process. Because of the high adjacency of traces of differing propagation modes, an erroneous connection of the traces has occurred and the assumption that one trace structure corresponds to a single propagation mode has been violated. Furthermore, loops are present that will be difficult for the simple trimming approach presented above to handle. (In fact, the trace segment of figure 3.12 is an enlargement of the end of the trace depicted in figure 3.16.) This trace clearly shows the need for a more sophisticated approach to trimming.

The new approach I propose proceeds in the following manner. Firstly, if the maximum length of the trace structure is less than a critical threshold, then we delete it just as before. Secondly, if it is long enough to consider, then we construct a piece-wise linear approximation to the thinned structure as follows. Between every vertex (including terminal points) of the thinned structure, we draw a line segment. If any pixel in a branch of the thinned structure is more than the width of two pixels away from the line segment that contains the branch's vertices, then consider one of these distant pixels to be an imaginary vertex of an additional branch and break the line segment into two. If there are still more pixels over two pixel widths from the new line segments, then add another imaginary vertex, shifting them both if necessary (figure 3.17). Repeat these steps if necessary.

After an accurate (to two pixel widths) piece-wise linear approximation to the thinned structure has been completed, the next step is to delete any pixels that do not contribute to the shortest 8-connected path between its bounding vertices. This will remove small loop structures from the thinned ionogram as is shown in figure 3.18. Furthermore, any small loop caused by 8-connected vertices also indicate the presence of pixels that can be removed (figure 3.19). Sometimes there may be a choice between possible pixels that could be removed, and when this occurs delete those that are furthest from the linear segment.

[2]Since writing this portion of the report, an improved thinning algorithm based on these ideas has been developed by David Kettler [16] and the results of this routine are incorporated in chapter 7.

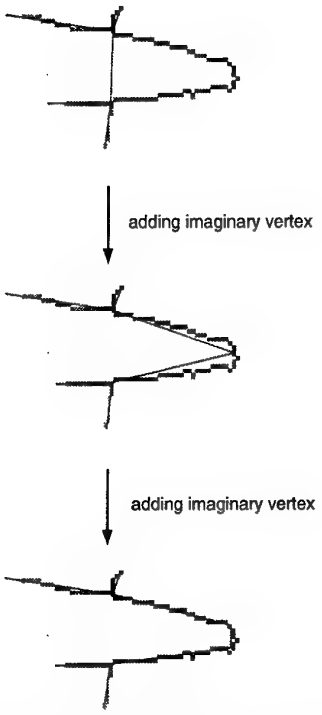


Figure 3.17: The procedure for adding imaginary vertices to the piece-wise linear approximation to the thinned trace structure.

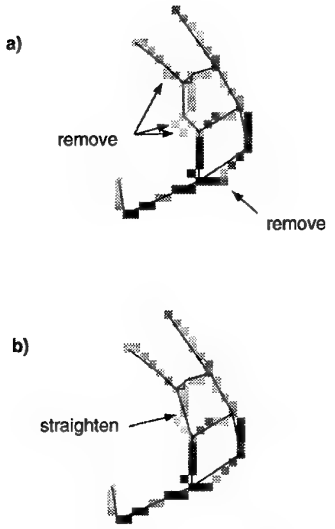


Figure 3.18: Removing pixels that do not contribute to the shortest 8-connected path between bounding vertices has the effect of removing small loops or holes from the thinned trace structure as in (a). When this has been performed, it is sometimes possible to remove vertices and "straighten out" the approximation as in (b).

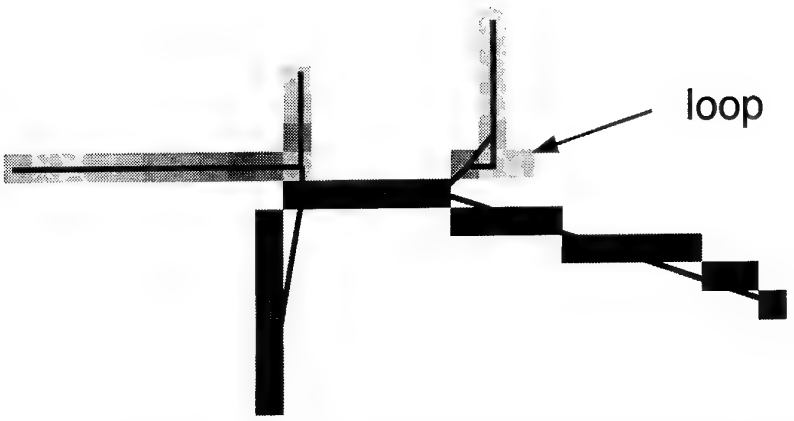


Figure 3.19: The presence of small loops due to a number of adjacent vertices indicates the presence of pixels that can be removed.

Alternatively, in this situation, do not delete those pixels that offer the best possibility for straightening up the piece-wise linear approximation across the neighbouring segments (*i.e.,* preserve the pixels that are closest to a "straightened out" line segment that would be formed if the nearest vertex to these pixels was deleted). After removing these pixels, see if it is now possible to remove some vertices from the piece-wise linear approximation in the neighbourhood of the removed pixels such that the two pixel error condition is still satisfied for the remaining pixels. This will have the effect of straightening out the approximation (see figure 3.18(b)).

Next, delete any short (below a critical threshold) terminal branch that is approximately perpendicular to other branches in its neighbourhood by testing the angle of intersection of the linear segments. Further, if any non-terminal branch is perpendicular to the traces it connects, which are roughly parallel, then delete it. This will have the effect of removing the "steps" from the "ladder" structures in the thinned trace. Next, cut non-terminal branch structures at the point that they are perpendicular to long series of roughly straight line segments. This will have the effect of separating out branches that are unlikely to be of the same propagation mode as the long straight segments, and will make branch selection much easier.

These steps have been performed by hand upon the difficult trace of figure 3.16. The results are shown in figure 3.20. Note that in this figure these steps have also had the effect of breaking the trace down into separated traces that more closely correspond to a single propagation mode, and so has made branch selection in this case straightforward.

The steps involved in the improved trimming algorithm for each trace are summarized as follows:

1. If the length of the longest pathway through the thinned structure is less than a critical threshold, then delete it, and terminate the processing for this trace.

2. Construct a piece-wise linear approximation to the thinned trace structure thus:

   (a) Identify all pixels that are vertices (those that have 3 or more neighbours or are terminal points).

   (b) Draw a line segment between adjacent vertices.

   (c) If any pixel is more than 2 pixel widths away from its corresponding line segment, then add imaginary vertices one at a time, moving them as necessary, until all pixels are close enough.

3. Delete pixels that do not contribute to the shortest 8-connected path between its bounding vertices. If there is a choice, delete the one that is furthest from its corresponding line segment, or keep

Figure 3.20: The advanced procedure for trimming a thinned ionogram trace. a) Piecewise-linear approximation to the trace structure. b) Final result.

pixels that are closest to the line segment that would be formed if the closest vertex was deleted.

4. Remove pixels indicated by small loops that result from 8-connected vertices.
5. Of those vertices having only two neighbouring connected vertices, see if it is possible to demote any of them to pixels and still satisfy the 2 pixel error condition.
6. Delete short terminal branches that are approximately perpendicular to the line segments that they join.
7. Delete non-terminal perpendicular branches between approximately parallel segments.
8. Cut non-terminal branch structures at the point that they are perpendicular to long series of roughly straight line segments.

The thinned filtered test ionograms have been hand-trimmed using this algorithm and the results can be seen in figures 3.34–3.37.

3.7 Improved Branch Selection

After the improved trimming of a trace structure, the branch selection problem is much easier. In fact, the branches selected by longest pathway in trace structures hand trimmed according to the improved algorithm can be seen in figures 3.38–3.41, and no problems are evident in these cases. However, there are still some difficulties that it may occasionally have to cope with, so we still need to suggest something more sophisticated than the "longest pathway" approach if we want to have the greatest chance of success in dealing with an ionogram. The first problem must cope with relates to the separation of x- and o-rays, which are features that sometimes occur at the high frequency end of an ionogram trace.

3.7.1 Separation of X and O Rays

The LLISP ionograms upon which I have developed this autoscaling system do not strongly exhibit a phenomenon called *magneto-ionic splitting*, which can cause the upper ray of traces (the portion that points up and to the left and is sometimes missing from a trace) to have a "forked tongue" (see figure 3.21 for an example of a trace with this behaviour). The left-hand branch of the forked tongue is called the

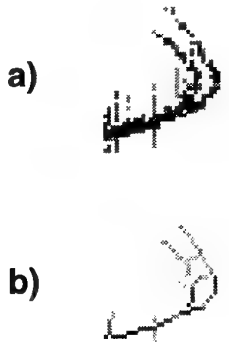


Figure 3.21: The "forked tongue" effect of magneto-ionic splitting.

o-ray and the right-hand one the x-ray. This forked tongue effect is much more significant in oblique ionograms such as the FMS ionograms collected by High Frequency Radar Division.

In ionogram traces where there is magneto-ionic splitting, it is important to ensure that one ray is selected by the branch selection scheme, and currently no special provision is made to cater for these occurrences in my system. However, a solution to the problem of separating the x- and o-rays has been developed by Dr Matthew Roughan in his system for trace extraction [17], and his approach could easily be integrated into my system. The most appropriate place for this integration to occur is in the branch selection stage. By doing this, the autoscaling system presented here would be applicable to an even

broader class of oblique ionograms. Special treatment of this problem is important in ionograms where the thinning algorithm has failed to separate the rays – see figure 3.9 for an example where this has occurred.

3.7.2 Selecting "Smooth" Branches

The improved thinning algorithm makes the problem of selecting the branches for fitting much easier. However, we can still improve on the longest pathway branch selection scheme already mentioned. Visually, an ionogram trace corresponding to a single propagation mode is recognizable because it is a smooth curve. Thus a better approach to branch selection is to selected the longest smoothly bending pathway through the branch structure, and this could be efficiently determined from the piece-wise linear approximation to the trace already computed by the improved trimming algorithm.

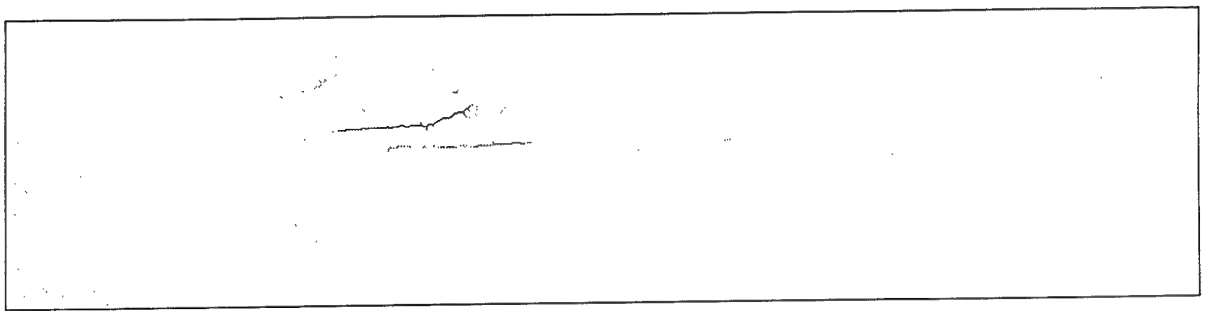


Figure 3.22: Ionogram 1 — thinned.

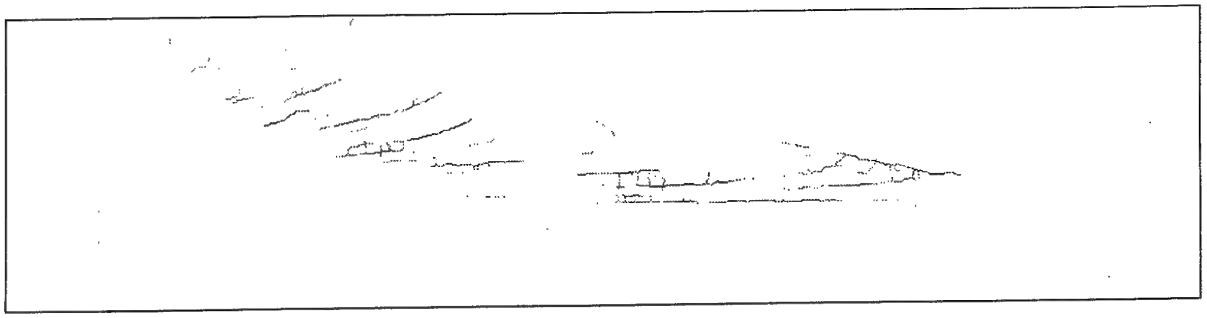


Figure 3.23: Ionogram 2 — thinned.

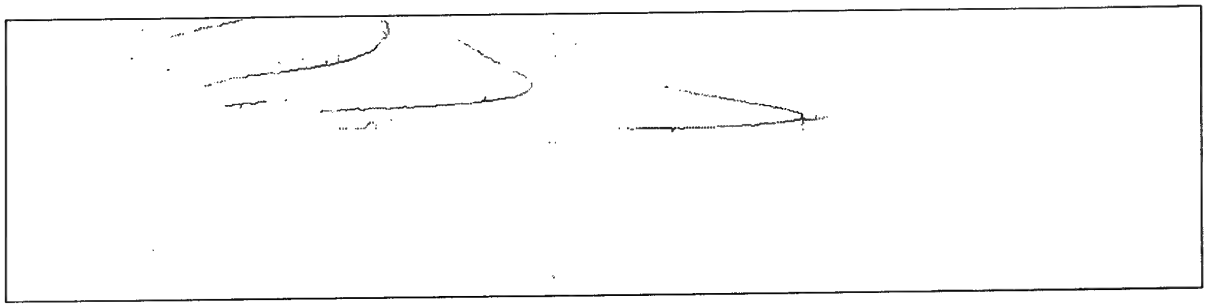


Figure 3.24: Ionogram 3 — thinned.

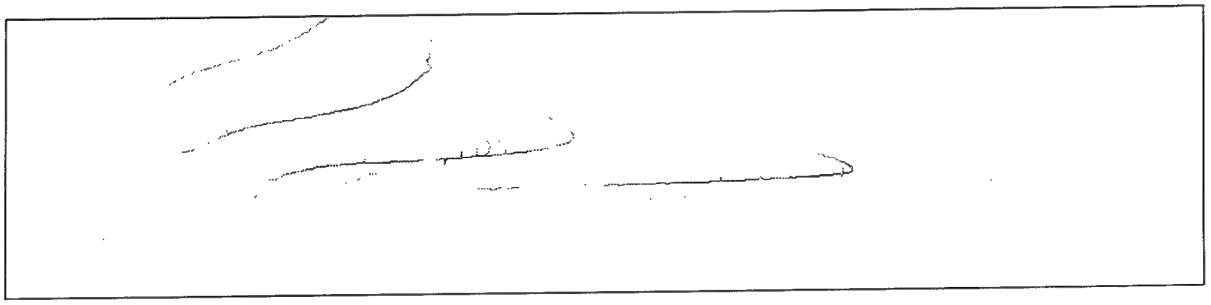


Figure 3.25: Ionogram 4 — thinned.

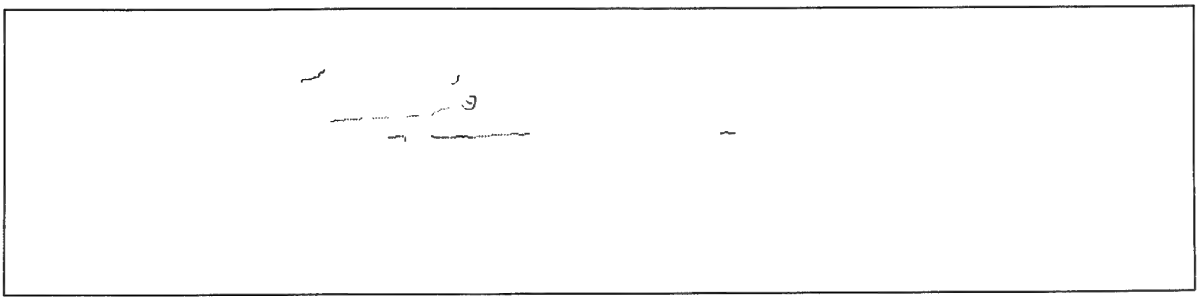


Figure 3.26: Ionogram 1 — trimmed and thinned.

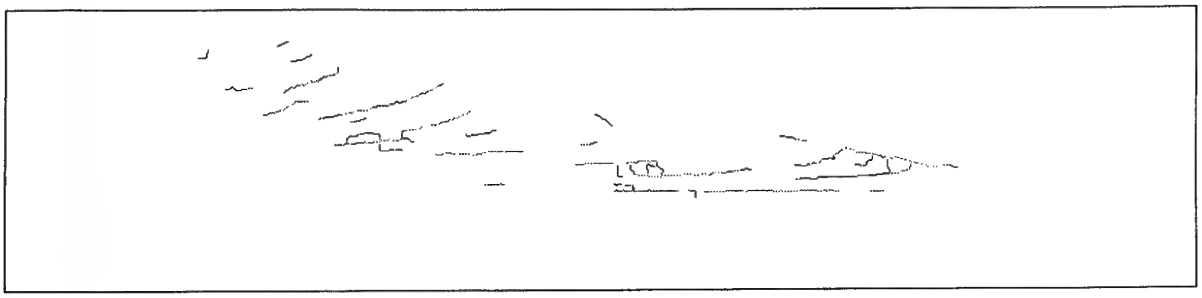


Figure 3.27: Ionogram 2 — trimmed and thinned.

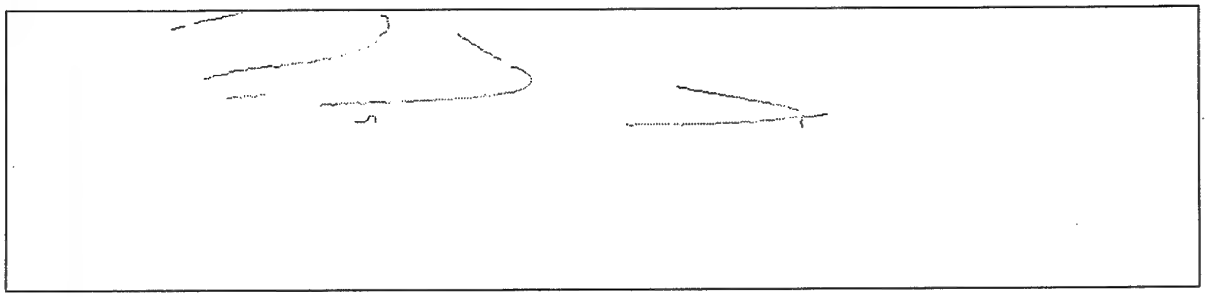


Figure 3.28: Ionogram 3 — trimmed and thinned.

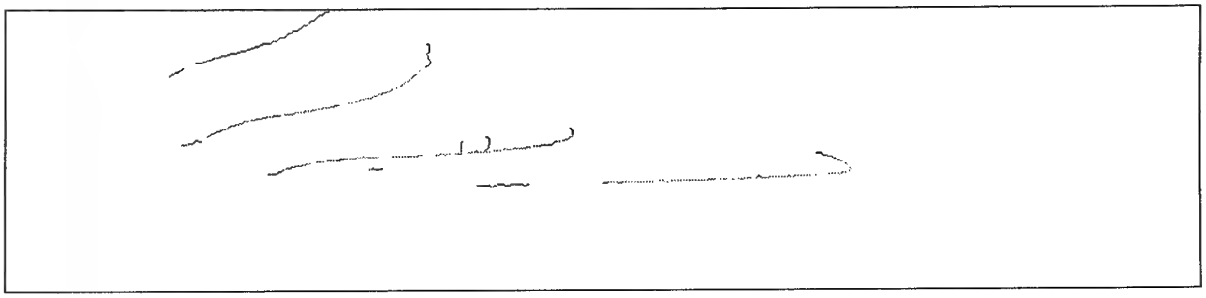


Figure 3.29: Ionogram 4 — trimmed and thinned.

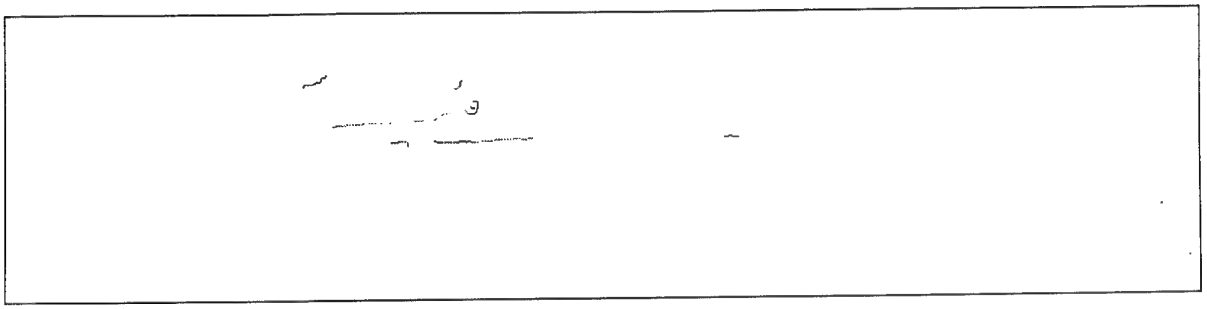


Figure 3.30: Ionogram 1 — selected branches.

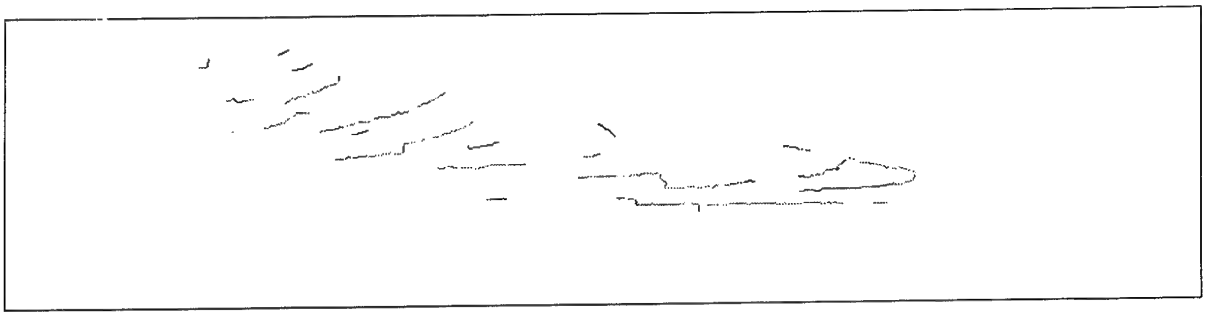


Figure 3.31: Ionogram 2 — selected branches.

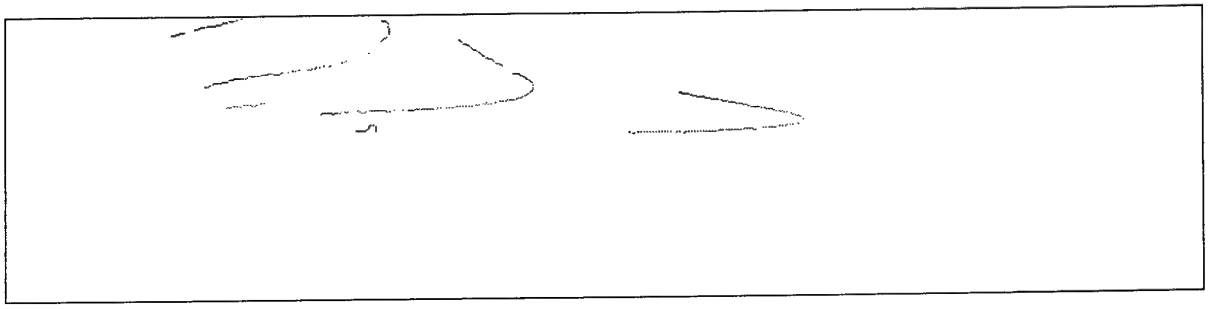


Figure 3.32: Ionogram 3 — selected branches.

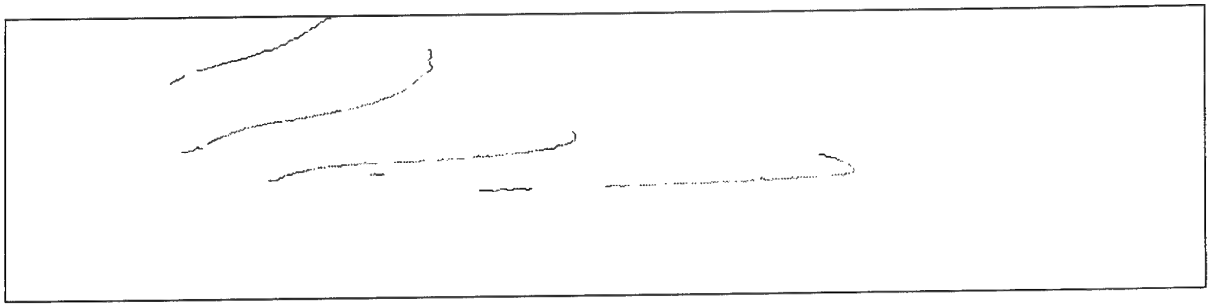


Figure 3.33: Ionogram 4 — selected branches.

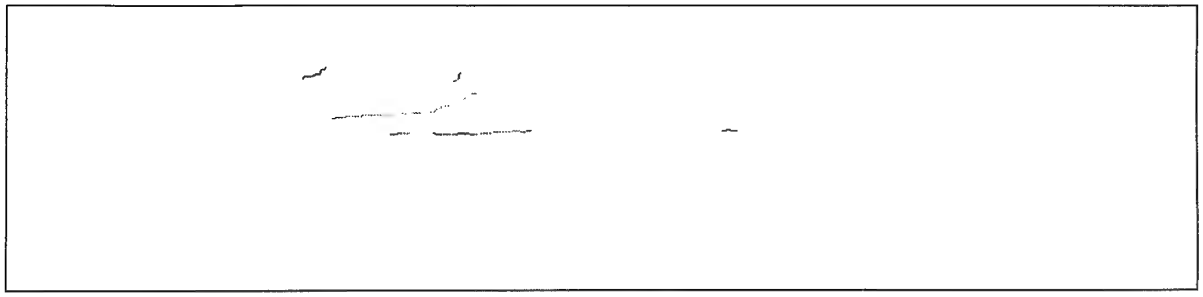


Figure 3.34: Ionogram 1 — hand trimmed by improved algorithm.

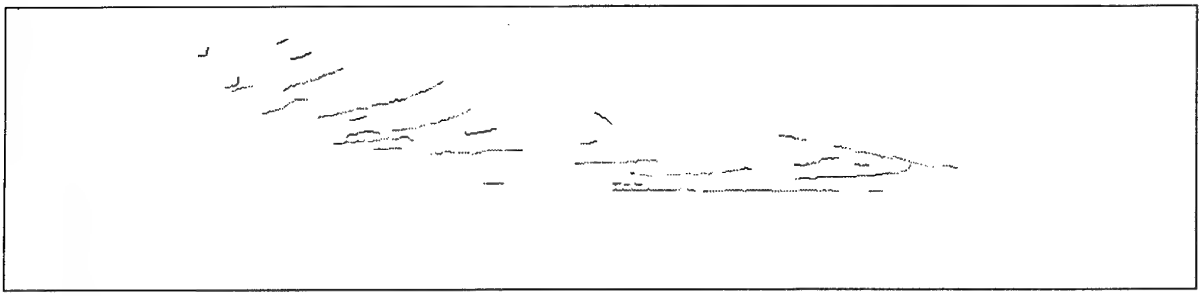


Figure 3.35: Ionogram 2 — hand trimmed by improved algorithm.

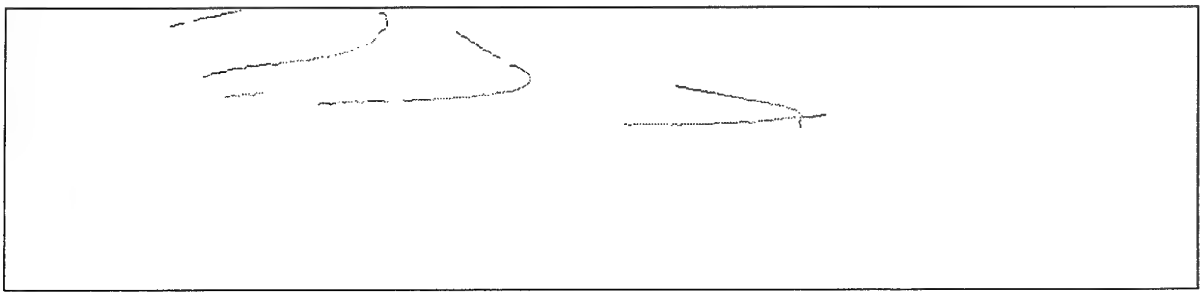


Figure 3.36: Ionogram 3 — hand trimmed by improved algorithm.

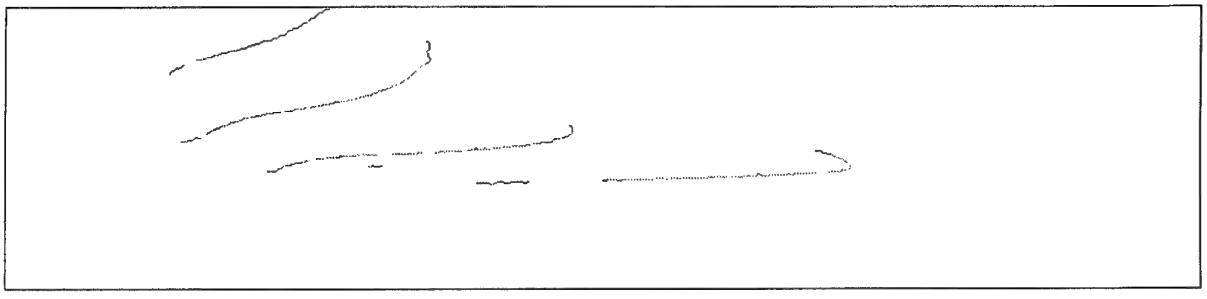


Figure 3.37: Ionogram 4 — hand trimmed by improved algorithm.

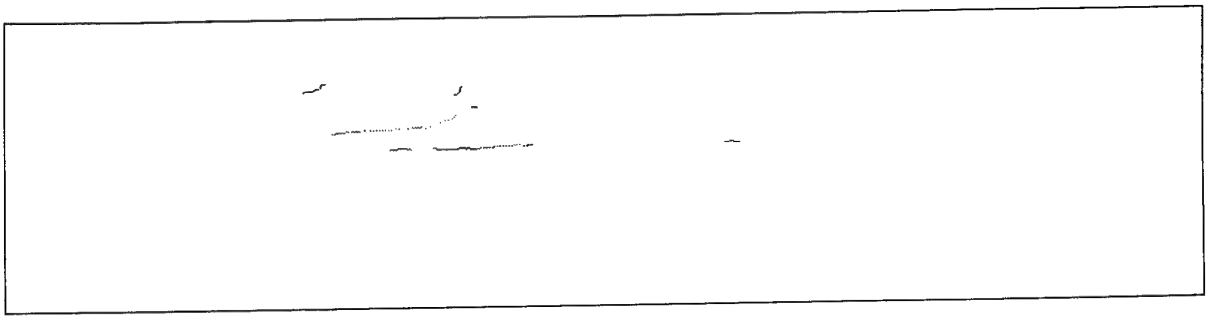


Figure 3.38: Ionogram 1 — selected branches of hand trimmed traces, trimmed according to improved algorithm.

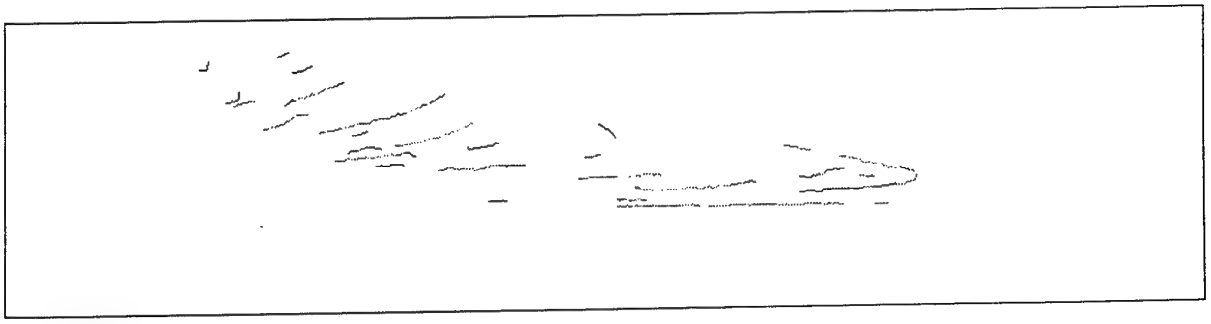


Figure 3.39: Ionogram 2 — selected branches of hand trimmed traces, trimmed according to improved algorithm.

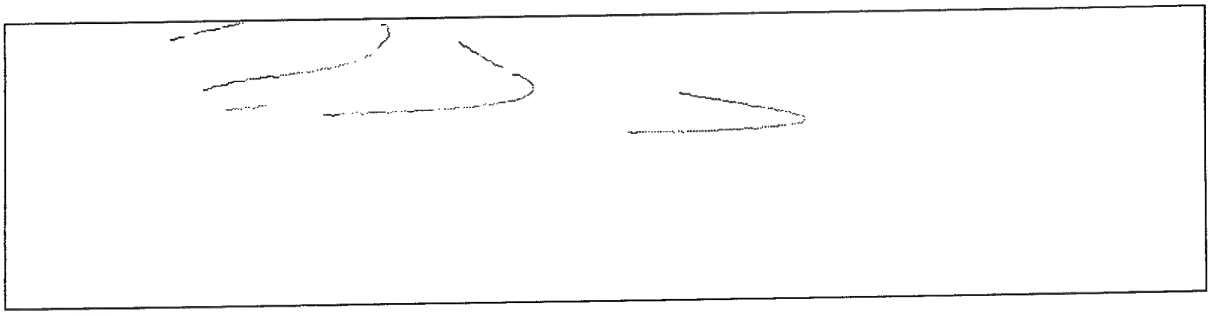


Figure 3.40: Ionogram 3 — selected branches of hand trimmed traces, trimmed according to improved algorithm.

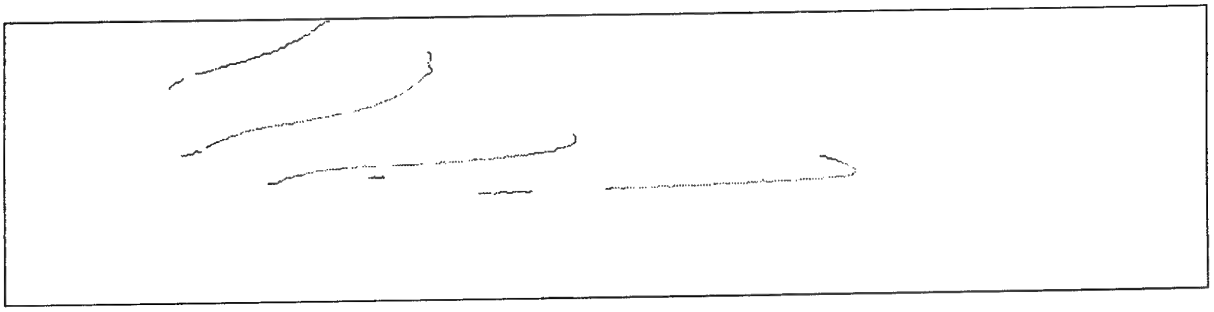


Figure 3.41: Ionogram 4 — selected branches of hand trimmed traces, trimmed according to improved algorithm.

Chapter 4

Feature Extraction: Fitting

The next stage of the feature extraction process after skeletonization, trimming and branch selection centres on finding a small number of parameters that efficiently describe the shape of the selected branches in the thinned ionograms. This will be done by *fitting* a model shape to the thinned-trimmed traces, *i.e.*, by determining the best parameters such that a model's shape is closest to the given trace. The resulting feature vector is then passed on to subsequent processing stages (to be described in forthcoming chapters) which will identify disjoint pieces of the same underlying trace and merge them, and track their evolution over sequences of ionograms. The results can then be used to identify ionogram traces with propagation modes.

It may be possible to perform feature extraction by fitting directly to the filtered ionogram traces (without first thinning and trimming them) using an optimization procedure similar to the one presented in this chapter. In some circumstances (where there are no confusing branches present) this could achieve a better fit for each trace, but this would be at a cost of greatly increased computation. This increase would be very significant, because the optimization procedure used for fitting is the most computationally intensive algorithm employed in the whole autoscaling system. I have already attempted fitting directly to the filtered traces using a number of Hough transform techniques (some of which were novel variations) with a variety of different trace models. The problems encountered by this methodology proved insurmountable and so these approaches were reluctantly abandoned[1].

There are two key aspects to fitting after the principal branches have been selected according to the methods presented in the previous chapter. They are: selecting a suitable model and finding a method of determining the best model parameters. These elements are depicted in figure 4.1.

4.1 Feature spaces and associated parameterizations

Two aspects of an ionogram trace are essential for characterizing for future matching, merging and tracking stages, and consequently identifying its propagation mode. They are: its location in the frequency / group-delay space of the ionogram, and its shape. Therefore we seek a quantitative description of these properties.

The output of the feature extraction process will be a single point, one for each trace, in a multi-dimensional space called the *feature space*. Let $f(\mathbf{x}, \boldsymbol{\beta})$ denote the trace model with model parameter $\boldsymbol{\beta}$ on the ionogram with pixel coordinates denoted by $\mathbf{x}$. The feature space is derived from model parameters $\boldsymbol{\beta}$ by adding to them details of the size and location of the trace (figure 4.2). As a consequence, the choice of model determines the characteristics of the feature space, and we must carefully consider what model is most appropriate for the traces.

Let us firstly, though, consider what criteria we wish the feature space to satisfy, because the properties of the feature space will drive the choice of model. Therefore we have the following criteria for the feature space.

[1] These problems were due to the nature of the ionogram traces — they are not describable by any reasonable analytic model. Therefore, when a generalized Hough transform [18] approach is employed with such a model, the activity in the Hough space will be spread out across a range of cells. As a consequence, there is no peak in the Hough space that corresponds to values of the parameter model for which there is a best match of the model to the trace. Various techniques were tried to overcome this but to no avail.

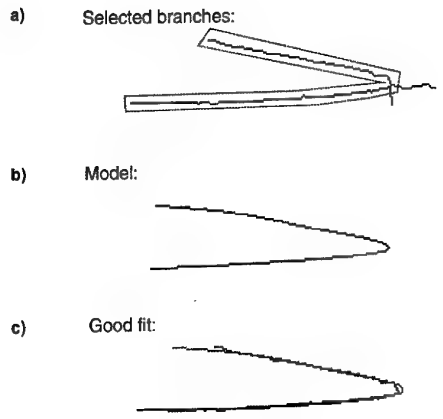


Figure 4.1: The elements of the fitting stage of feature extraction. The selected branches of a trimmed-thinned ionogram trace (a) is fitted to a model (b) so that for some value of the parameter (b) there is a good correspondence of the model to the selected traces as shown in (c).

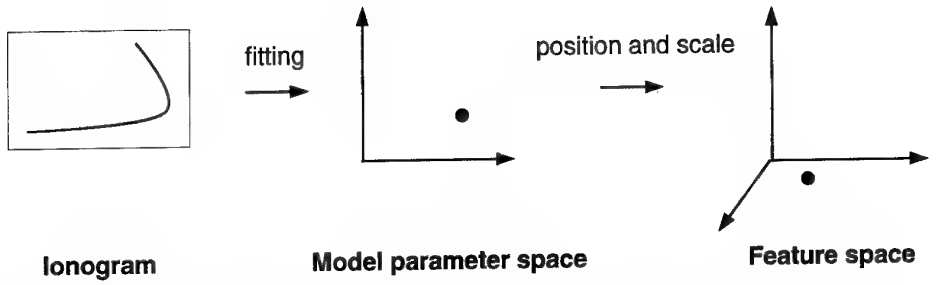


Figure 4.2: The relationship between the ionogram, its model, and the feature space.

Criterion 1 The feature space must have a small number of dimensions.

Criterion 2 The feature space must encode information on the position, scale and shape of traces and these will preferably be independent dimensions of the feature space.

Criterion 3 The mapping from trace to feature vector should, as far as is possible, preserve distances: shapes that are close should map to feature vectors that are close; while features that are close should produce shapes that are close when they are replotted in the original space.

The need for the first criterion to be satisfied by the feature space is obvious: a low dimensional representation of a trace will produce short feature vectors and so reduce the data that is passed on to further processing stages. Ideally this reduced data set will still retain all the information relevant to any further processing and simply be representing it in a more abstract form. This goal is the basis of the remaining two criteria.

Criteria two and three above are motivated by the need to know the relative positions and shapes of different traces so that in the end their propagation modes can be determined. If the autoscaling system is to make sensible decisions as to whether two traces are the same or are different, distances between feature vectors must reflect natural measures of the distances between the shape of traces. For example, if a small change in the shape of a trace produces an exponential change in the model's best fit, then this criterion would be poorly met. Traces corresponding to a particular mode can move and evolve between successive ionograms; the autoscaling system must be able to decide whether a trace is a translation of a similar shape in a previous trace or not. Furthermore, this allows interesting possibilities such as classification of traces by their shape so that incompletely understood ionospheric phenomena that manifest themselves as perturbations in trace shape can be identified. We will consider these ideas in more detail later.

The three criteria above dictate characteristics that the model must have. The first characteristic follows immediately from the first criterion above.

Characteristic 1 The model must have few parameters.

To be able to compare traces by their feature vectors (the reasoning behind criteria two and three above) we need to be able to describe their shape irrespective of position using the model parameters. Therefore, it must be possible to translate and resize the model using a simple transformation of the model parameters. Consequently, the model must be invariant under magnification and translation. Moreover as changes of size and location will be common when attempting to merge trace segments, it seems reasonable to require that the best fit to a resized and translated set of data should be easily derived from the best fit to the original data by a simple *linear* transform of the parameters defining the best fit to the original data. (Note that this implicitly restricts the fitting metric as well as the family of curves.) Therefore, the second model characteristic is:

Characteristic 2 It must be possible to translate and resize the trace by a simple linear transformation of the model parameters.

As noted previously, interference, noise, insufficient antenna gain and other effects can break traces, so that an ionogram may have two or more distinct traces that are nevertheless clearly associated with the one underlying structure. These structures should manifest themselves as smooth curves in ideal ionograms, so merging can be carried out by smoothly extrapolating trace segments and checking for overlaps. This is termed *merging*. Thus:

Characteristic 3 The model must allow extrapolation past the end of an existing trace in a manner consistent with the general trend of the trace.

Lastly, because we want the model to have as few parameters as possible, and at the same time we want it to smoothly follow the general shape of the trace we must make a compromise between the two. That is, we want the trace to follow the general shape of the trace without being too complicated due to following every minor detail. Yet, at the same time, the model must be a reasonable representation of the trace. The degree to which we compromise between the two will be determined by the needs of future processing stages. Now, the eye is an excellent judge of the shape of the traces, and so we use it as the ultimate arbiter in this decision. That is, if it appears that the model doesn't follow trace behaviour to the human eye, then we must choose a better model. Therefore the last characteristic that the model must have is:

Characteristic 4 The model must be flexible enough to follow the usual behaviour of traces, yet restrictive enough that abnormal behaviour is highlighted.

The next section will present the development of a model that has these characteristics.

4.2 Choosing a Model and Metric

We need to choose a parameterization family $\mathcal{F}$ of curves $\mathcal{C}$ that meet the requirements for the trace model. As ionogram traces in general look like smooth curves, it is sensible to consider polynomials as candidates for our model. In fact, it seems likely that polynomials parameterized as linear subspaces are the only family of curves that could satisfy these requirements. There appear to be two possible ways of defining the family of curves.

Firstly, we could define the family by

$$\mathcal{F} = \bigcup_{\boldsymbol{\beta}} \mathcal{C}_{\boldsymbol{\beta}} \tag{4.1}$$

where the curves are given by

$$\mathcal{C}_{\boldsymbol{\beta}} \equiv \{\mathbf{x} : f(\mathbf{x}, \boldsymbol{\beta}) = 0\} \tag{4.2}$$

and

$$f(\mathbf{x}, \boldsymbol{\beta}) \equiv \sum_{i=0}^{P} \sum_{j=0}^{Q} \beta_{ij} x^i y^j. \tag{4.3}$$

Note that $\mathbf{x}$ will be used to denote the vector $\mathbf{x} = (x, y)$ where x and y represent frequency and group-delay, respectively. The vector $\boldsymbol{\beta}$ will denote the parameter vector defining a curve $\mathcal{C} \equiv \mathcal{C}_{\boldsymbol{\beta}}$; its components will be denoted by β_i or β_{ij}.

Secondly, we could instead define

$$\mathcal{F} = \bigcup_{\boldsymbol{\beta}} \mathcal{C}_{\boldsymbol{\beta}}$$

where

$$\mathcal{C}_{\boldsymbol{\beta}} \equiv \{(x(t), y(t)) : x(t) = \sum_{i=0}^{P} \beta_{1i} t^i, y(t) = \sum_{j=0}^{Q} \beta_{2j} t^j\}.$$

However, this parametric approach has many difficult problems relating to determining a suitable parameter t. For example, if arc length is used, then suitable starting points must be chosen and recorded. Therefore we will use the parameterization found in (4.3).

The model (4.1,4.2,4.3) can be made to have the first characteristic by a suitably small choice of P and Q. Furthermore, any model of this type has the ability to extrapolate past the data points to which it was fitted in a smooth manner, so it has characteristic 3. The model will have the fourth characteristic again if a suitable choice of P and Q is made. We will show that it can be made to have the second characteristic in subsection 4.2.2 to follow.

4.2.1 Metrics for Curve Fitting

Given the form of the model, the next issue to be addressed is which metric should be used to determine how well the model fits the trace.

Let $\mathcal{S} = \{\mathbf{x}_n\}_{n=1}^{N}$ be the set of data points to which we wish to fit a particular curve $\mathcal{C}$ of the model $\mathcal{F} = \cup_{\boldsymbol{\beta}} \mathcal{C}_{\boldsymbol{\beta}}$. To do so, we need to define a metric $d(\mathcal{C}, \mathcal{S})$ for the goodness of fit of $\mathcal{C}$ to $\mathcal{S}$: once d has been chosen we then choose a particular curve $\mathcal{C} \equiv \mathcal{C}_{\boldsymbol{\beta}}$ by solving

$$\min_{\boldsymbol{\beta}} d(\mathcal{C}_{\boldsymbol{\beta}}, \mathcal{S}).$$

The properties that we would like d to have are:

- The best fit to a rotated data set should be the rotation of the best fit to the original data.
- The best fit to magnified and translated data should be the magnified and translated version of the best fit to the original data.

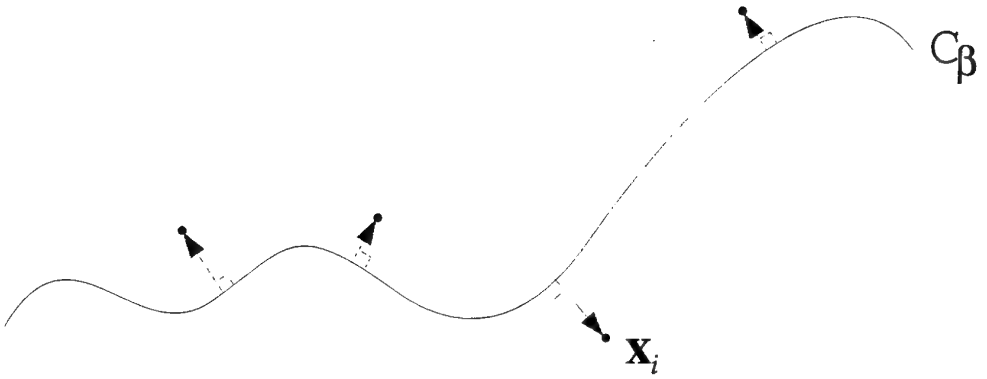


Figure 4.3: The metric used for the curve fitting operation is a sum of squared Euclidean distances of the points from the curve.

Therefore, a possible candidate for metric d is the standard distance metric as follows.

Since the distance $d(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^2$ is invariant under rotation, translation and magnification of the data, then $d(\mathcal{C}_{\boldsymbol{\beta}},\mathcal{S})$ also has these properties. Therefore, we define $d(\mathcal{C}_{\boldsymbol{\beta}},\mathcal{S})$ to be the sum of the squared Euclidean distances of the set of points $\mathcal{S}$ from the curve $\mathcal{C}_{\boldsymbol{\beta}}$ by

$$d(\mathcal{C}_{\boldsymbol{\beta}},\mathcal{S}) = \sum_{n=1}^{N} \min_{\mathbf{x}\in\mathcal{C}_{\boldsymbol{\beta}}} \|\mathbf{x}_n - \mathbf{x}\|^2. \tag{4.4}$$

In other words, we are attempting to minimize the distance between each data point in the set $\mathcal{S}$ and the closest point to it on the curve $\mathcal{C}_{\boldsymbol{\beta}}$ by varying $\boldsymbol{\beta}$. This is depicted in figure 4.3.

The disadvantage of this metric is that the function $\mathbf{x} = \arg\min_{\mathbf{y}\in\mathcal{C}_{\boldsymbol{\beta}}} \|\mathbf{x}_n - \mathbf{y}\|^2$ can be a discontinuous function of the data point $\mathbf{x}_n$ if $\mathcal{C}_{\boldsymbol{\beta}}$ is curved. Thus $d(\mathcal{C}_{\boldsymbol{\beta}},\mathcal{S})$ may have discontinuities in its first and higher derivatives. Also the metric is not consistent with any reasonable statistical model for how the data is formed, for example from knowledge that the data points recorded are formed by random perturbation of points on the underlying curve. In fact, in the case of errors in both x and y, the least squares estimate is a biased estimator even when fitting straight lines [19]. An alternative model that is motivated by a statistical model of formation of data could be developed (see [20] for the issues) but would require a lot of additional work.

4.2.2 Change of Coordinates

Let us now examine the behaviour of the model (4.1,4.2,4.3) under the change of size and position

$$\mathbf{x}' = \alpha(\mathbf{x}-\mathbf{u}) \tag{4.5}$$

where $\mathbf{u}$ denotes the shift in origin and α the magnification parameter.

If $\mathcal{C}_{\boldsymbol{\beta}(1,0)}$ denotes the best fit in the original coordinate system and $\mathcal{C}_{\boldsymbol{\beta}(\alpha,\mathbf{u})}$ denotes the best fit in the new coordinate system after applying (4.5), then

$$\mathbf{x} \in \mathcal{C}_{\boldsymbol{\beta}(1,0)} \quad \Longleftrightarrow \quad \mathbf{x}' = \alpha(\mathbf{x}-\mathbf{u}) \in \mathcal{C}_{\boldsymbol{\beta}(\alpha,\mathbf{u})}$$

and so

$$f(\mathbf{x},\boldsymbol{\beta}(1,0)) = 0 \quad \Longleftrightarrow \quad f(\mathbf{x}',\boldsymbol{\beta}(\alpha,u)) = 0.$$

This follows from

$$f(\mathbf{x},\boldsymbol{\beta}(1,0)) \;=\; \sum_{i=0}^{P}\sum_{j=0}^{Q} \beta_{ij} x^i y^j$$

$$
\begin{aligned}
&= \sum_{i=0}^{P}\sum_{j=0}^{Q}\beta_{ij}(\frac{x'}{\alpha}+u)^i(\frac{y'}{\alpha}+v)^j \\
&= \sum_{i=0}^{P}\sum_{j=0}^{Q}\beta_{ij}\sum_{r=0}^{i}\binom{i}{r}\left(\frac{x'}{\alpha}\right)^r u^{i-r}\sum_{s=0}^{j}\binom{j}{s}\left(\frac{y'}{\alpha}\right)^s v^{j-s} \\
&= \sum_{r=0}^{P}\sum_{s=0}^{Q}x'^r y'^s\frac{1}{\alpha^{r+s}}\sum_{i\geqslant r}^{P}\sum_{j\geqslant s}^{Q}\binom{i}{r}u^{i-r}\binom{j}{s}v^{j-s}\beta_{ij} \\
&= f(\mathbf{x}', \boldsymbol{\beta}(\alpha, u)).
\end{aligned}
$$

This gives the resulting lemma.

Lemma 1 *Under the coordinate change* $\mathbf{x}' = \alpha(\mathbf{x} - \mathbf{u})$, *the curve of best fit transforms from* $\mathcal{C}_{\boldsymbol{\beta}}$ *to* $\mathcal{C}_{\boldsymbol{\beta}'}$ *where*

$$\boldsymbol{\beta}' = \mathbf{A}\boldsymbol{\beta}$$

with

$$A_{rs,ij} = \begin{cases} \frac{1}{\alpha^{r+s}}\binom{i}{r}u^{i-r}\binom{j}{s}v^{j-s} & i \geqslant r, j \geqslant s \\ 0 & \textit{otherwise.} \end{cases}$$

Furthermore, the metric defining the best fit transform is

$$d(\mathcal{C}_{\boldsymbol{\beta}(\alpha,u)}, \mathcal{S}') = \alpha^2 d(\mathcal{C}_{\boldsymbol{\beta}(1,0)}, \mathcal{S}).$$

Note, though, that a parameter vector $\boldsymbol{\beta}$ only defines $\mathcal{C}_{\boldsymbol{\beta}}$ uniquely up to arbitrary scaling of $\boldsymbol{\beta}$. Therefore it is permissible to arbitrarily rescale the matrix $\mathbf{A}$ above. In particular one reasonable choice is to rescale $\mathbf{A}$ so that $\det(\mathbf{A}) = 1$. Since $\mathbf{A}$ is upper triangular, $\det(\mathbf{A})$ is just the product of the diagonal elements. This gives

$$A_{rs,ij} = \begin{cases} \frac{\alpha^{\sum_{k=0}^{P}\sum_{l=0}^{Q}(k+l)/(P+1)(Q+1)}}{\alpha^{r+s}}\binom{i}{r}u^{i-r}\binom{j}{s}v^{j-s} & i \geqslant r, j \geqslant s \\ 0 & \text{otherwise.} \end{cases} \tag{4.6}$$

Thus we see that the best fitting curve remains essentially unchanged under a change (4.5) and in the fitting metric $d(\mathcal{C}_{\boldsymbol{\beta}}, \mathcal{S})$ defined by

$$d(\mathcal{C}_{\boldsymbol{\beta}}, \mathcal{S}) \equiv \sum_{n=1}^{N}\min_{\mathbf{x}\in\mathcal{C}_{\boldsymbol{\beta}}}\|\mathbf{x}_n - \mathbf{x}\|^2$$

and the model given by the family $\mathcal{F}$ of parameterized curves $\mathcal{C}_{\boldsymbol{\beta}}$, specified by

$$\mathcal{C}_{\boldsymbol{\beta}} \equiv \{(x,y) : f(\mathbf{x} : \boldsymbol{\beta}) = \sum_{i=0}^{P}\sum_{j=0}^{Q}\beta_{ij}x^i y^i = 0\}.$$

Furthermore, the transformation is linear, so the model (4.1,4.2,4.3) has characteristic 2.

Chosen Model

The complete trace model that I chose is

$$\mathcal{C}_{\boldsymbol{\beta}} \equiv \{\mathbf{x} : f(\mathbf{x}, \boldsymbol{\beta}) = 0\} \tag{4.7}$$

where the function f is quartic in y and linear in x, given by

$$f(\mathbf{x}, \boldsymbol{\beta}) \equiv \sum_{i=0}^{4}\beta_i y^i + \beta_5 x. \tag{4.8}$$

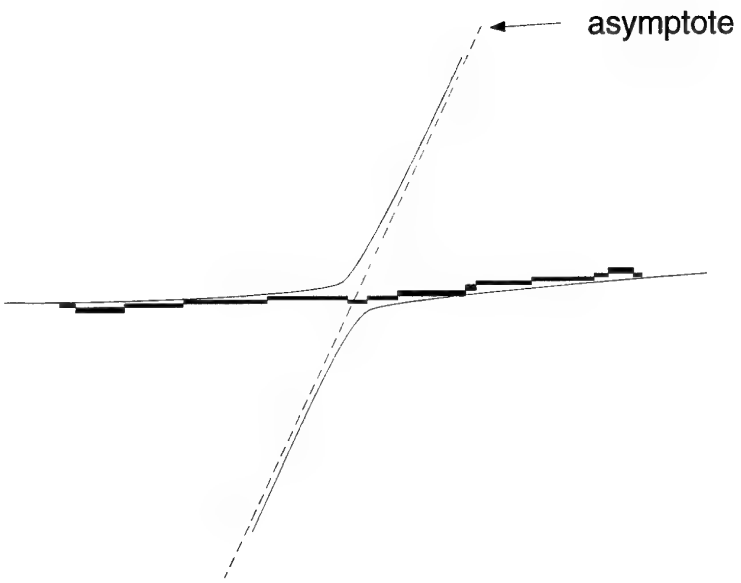


Figure 4.4: Bad placement of asymptotes.

This choice was made because traces are approximately[2] x is a function of y, with the addition that y is sometimes constant (*i.e.*, a trace may be a long straight line roughly parallel to the frequency axis).

Simpler models were tried that employed quadratics and cubics in y with and without an implicit x term. It was found that these did not have the ability to track the trace around the nose region closely enough, or have enough "bends" to follow the upper and lower-angle rays of traces.

4.2.3 Other Models Tried and Rejected

We now review some of the various models for traces that were considered and rejected in deciding on a suitable feature vector.

The first model was based on the observation that the majority of ionogram traces are essentially smooth "corners". That is, they are composed of a bend extending out to two "straight" sections (one or both of which may be absent). These features would seem to be well fitted by a generalized hyperbola: the two straight sections would be modelled by the asymptotes and the remainder of the hyperbola would form the corner. The hyperbola was parameterized using the implicit model

$$\begin{aligned} f(x,y:\boldsymbol{\beta}) &= \cos\theta_1\cos\theta_2(y-y_0)^2 - \sin(\theta_1+\theta_2)(x-x_0)(y-y_0) \\ &\quad + \sin\theta_1\sin\theta_2(x-x_0)^2 - k\sin^2(\theta_1-\theta_2) \\ &= 0 \end{aligned}$$

where the parameter vector $\boldsymbol{\beta} = (x_0, y_0, \theta_1, \theta_2, k)$ consists of the location (x_0, y_0) where the two asymptotes intersect, the angles θ_1 and θ_2 that the two asymptotes make with the x-axis, and the constant k defining how close the hyperbola is to the asymptotes.

This model proved to be unsatisfactory because of unreliable placements of the asymptotes. Sometimes the intersection of the asymptotes would be located in the middle of a roughly straight trace as depicted in figure 4.4. This poor placement can nevertheless still provide a good fit through aligning one of the asymptotes with the trace and then choosing a very small constant k to force the hyperbola to be close to the asymptotes. The resulting hyperbola is close to all points on the trace, but the associated model is a misleading indication of the trace's shape.

[2]The ionograms that were used to develop this model were taken during a period of high sun-spot number, where the underlying layers of the ionosphere have only a small effect. However, during periods of low sun-spot number this assumption is frequently violated and a different model is needed — see chapter 7.

Similar problems with asymptote placement dogged several other promising models: for instance the general conic model,

$$\beta_1 x^2 + \beta_2 y^2 + \beta_3 xy + \beta_4 x + \beta_5 y + 1 = 0,$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_5)$, and the cubic model

$$\beta_1 x^2 y + \beta_2 xy^2 + \beta_3 x^2 + \beta_4 y^2 + \beta_5 xy + \beta_6 x + \beta_7 y + 1 = 0,$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_7)$. Therefore both were abandoned.

Other approaches investigated involved explicit models in which frequency was expressed as a function of group delay (*i.e.*, x is expressed as a function of y). One of the explicit models trialed was the rational polynomial

$$x = \frac{\beta_1 y^2 + \beta_2 y + \beta_3}{\beta_4 y^2 + \beta_5 y + 1}.$$

The motivation for this model was that it included horizontal asymptotes that should give a good model of the long straight end-segments of traces. This approach again ran into problem with asymptote placement and had to be also abandoned.

Finally, feature spaces of these particular forms are not, of course, the only ones possible. For example, feature vectors are often compiled from Fourier descriptors of shapes [11]. This approach was rejected here, however, because they work best with shapes that have closed boundaries and non-empty interiors and they cannot be easily extrapolated. Furthermore, many other types of models can be rejected because they only capture local information about the trace.

4.3 Fitting the Model

Given that the model, the family $\mathcal{F}$ of parameterized curves $\mathcal{C}_{\boldsymbol{\beta}}$, is specified by

$$\mathcal{C}_{\boldsymbol{\beta}} \equiv \{(x, y) : f(\mathbf{x} : \boldsymbol{\beta}) = 0\}$$

and the fitting metric $d(\mathcal{C}_{\boldsymbol{\beta}}, \mathcal{S})$ by

$$d(\mathcal{C}_{\boldsymbol{\beta}}, \mathcal{S}) \equiv \sum_{n=1}^{N} \min_{\mathbf{x} \in \mathcal{C}_{\boldsymbol{\beta}}} \|\mathbf{x}_n - \mathbf{x}\|^2$$

it remains to determine which parameter $\boldsymbol{\beta}$ corresponds to a curve $\mathcal{C}_{\boldsymbol{\beta}}$ that best fits a given trace $\mathcal{S}$. That is, given a trace consisting of the set of points $\mathcal{S} = \{\mathbf{x}_i\}$, $\mathbf{x}_i = (x_i, y_i)$, $i = 1, \ldots, n$, we wish to find parameter values $\boldsymbol{\beta}$ such that the curve $f(x, y : \boldsymbol{\beta}) = 0$ best fits the data. The function f is smooth, and usually the number of data points n is much larger than the number of model parameters, $|\boldsymbol{\beta}|$. Therefore, the fitting problem is the one of determining a $\hat{\boldsymbol{\beta}}$ such that

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} d(\mathcal{C}_{\boldsymbol{\beta}}, \mathcal{S}).$$

We can express the above fitting problem in the following way. Let $\tilde{\mathbf{x}}_i$ denote the closest (using Euclidean distance) point on the curve $f(\mathbf{x} : \boldsymbol{\beta}) = 0$ to point $\mathbf{x}_i$. Let $\boldsymbol{\gamma}_i = \tilde{\mathbf{x}}_i - \mathbf{x}_i$ be the distance between them.

With these definitions,

$$\|\boldsymbol{\gamma}_i\|^2 = \min_{\mathbf{x} \in \mathcal{C}_{\boldsymbol{\beta}}} \|\mathbf{x}_n - \mathbf{x}\|^2,$$

and the fitting problem can be expressed as

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \sum_{i=1}^{n} \|\boldsymbol{\gamma}_i\|^2 \\ \text{subject to} \quad & f(\mathbf{x}_i + \boldsymbol{\gamma}_i : \boldsymbol{\beta}) = 0 \end{aligned} \tag{4.9}$$

where $\|\cdot\|$ is the Euclidean norm. This problem is called orthogonal distance regression (ODR): it is nontrivial, but algorithms to solve it can be found in [21–23]. (The distance gets its name from the fact that the line $\tilde{\mathbf{x}}_i - \mathbf{x}_i$ is orthogonal to the tangent of f at $\tilde{\mathbf{x}}_i$).

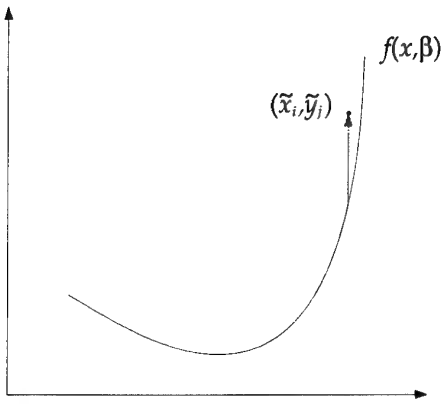


Figure 4.5: The distance measure of points from a curve in classical distance regression emphasizes the points along rapidly changing intervals of the curve.

Explicit Models

Some of the models that I tested were explicit functions: x was expressed as a function of y.

For the special case of fitting an explicit model, ODR reduces to an unconstrained optimization problem. In this case the orthogonal distance of a point (x_i, y_i) to the curve is now got by finding a point $(\tilde{x}_i, \tilde{y}_i)$ on the curve (*i.e.*, that satisfies $\tilde{x}_i = f(\tilde{y}_i : \boldsymbol{\beta})$) such that $(\gamma_i, \epsilon_i) = (\tilde{x}_i, \tilde{y}_i) - (x_i, y_i)$ is as small as possible. This can be written as:

$$\min_{\boldsymbol{\beta}} \quad \sum_{i=1}^{n} (\gamma_i^2 + \epsilon_i^2)$$

$$\text{subject to} \quad x_i + \gamma_i = f(y_i + \epsilon_i : \boldsymbol{\beta})$$

Since the constraint is linear in the γ_i, it can be eliminated to give the unconstrained optimization problem

$$\min_{\epsilon_i, \boldsymbol{\beta}} \quad \sum_{i=1}^{n} \left[(x_i - f(y_i + \epsilon_i : \boldsymbol{\beta}))^2 + \epsilon_i^2 \right]. \tag{4.10}$$

Again the algorithms in [21–23] include an option for ODR on explicit curves: this is more efficient than ODR on implicit curves as the minimization is now unconstrained.

Finally it is instructive to compare (4.10) with the standard nonlinear least squares method for estimating the parameter values $\boldsymbol{\beta}$. In this case only the observed dependent variable x_i is assumed to be in error. Thus the nearest point on the curve to (x_i, y_i) is the point $(x_i + \gamma_i, y_i)$ where

$$x_i = f(y_i : \boldsymbol{\beta}) - \gamma_i$$

where $\gamma_i \in \mathbb{R}$ is the error in observation x_i. If the errors are normally distributed with zero mean, then the maximum likelihood estimate $\hat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$ is the solution to the classical least squares problem

$$\min_{\boldsymbol{\beta}} \quad \sum_{i=1}^{n} \left[x_i - f(y_i : \boldsymbol{\beta}) \right]^2.$$

This gives a simpler minimization problem than (4.10), but at the price of giving undue weight to points in sections of the curve where the dependent variable is changing rapidly (figure 4.5).

4.4 Robust Fitting

The robustness of fit of the curve to the trace is greatly improved in three ways. They are, firstly, by performing the fitting in a normalized local coordinate system; secondly, by determining good initial guesses to the final parameter values; and lastly, by weighting the fit to each pixel of the trace according to its intensity. We will now consider these three aspects in detail.

4.4.1 Local Coordinates

Given an individual segment, we normalize its coordinate system for robustness' sake in the following fashion. The underlying principal is to transform the fitting problem to a local coordinate system in which the fitting problem should be reasonably well conditioned.

Note that the multivariate function describing the curves is chosen to be

$$f(\mathbf{x}, \boldsymbol{\beta}) = \sum_{i=0}^{4} \beta_i y^i + \beta_5 x.$$

Suppose that the set $\mathcal{S}$ of data points is constrained in the interval $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$. Then we rescale the data and curve fitting problem (preserving the aspect ratio) by moving to a new coordinate system such that

$$[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \quad \rightarrow \quad [-z, z] \times [0, 1],$$

where

$$z = \frac{x_{\max} - x_{\min}}{2(y_{\max} - y_{\min})}. \tag{4.11}$$

There is one proviso with this formula for z: if $y_{\max} - y_{\min} < 10$ then we set $y_{\max} - y_{\min} = 10$ in it. This is to ensure that the coverage of the local coordinate systems in y is never less than 10 pixels because with 4 degrees of freedom on this axis in the model we need to have some variation along it for numerical stability. Equation (4.11) then gives that

$$\mathbf{x}' = \frac{1}{y_{\max} - y_{\min}} (\mathbf{x} - (\frac{x_{\max} + x_{\min}}{2}, y_{\min})) \tag{4.12}$$

where $\mathbf{x}'$ is the value of $\mathbf{x}$ in the new coordinate system.

The reason for this normalization is to ensure that the constant β_0 in the best fitting polynomial over this region will almost certainly be comparable in size to the coefficient of higher order powers. This in turn allows us to replace the "correct" normalization

$$\sum_i \beta_i^2 = 1$$

by the *ad-hoc*, but easier to work with, normalization

$$\beta_0 = 1.$$

Thus, the trace model is the curve defined by

$$\mathcal{C}_{\boldsymbol{\beta}} \equiv \{(x, y) : f(\mathbf{x} : \boldsymbol{\beta}) = 1 + \sum_{i=1}^{4} \beta_i y^i + \beta_5 x = 0\}. \tag{4.13}$$

These normalizations are needed since the implicit definition of curves $\mathcal{C}$ in our chosen family $\mathcal{F}$ means that the curves are uniquely specified by a given parameter set only up to arbitrary scaling of that parameter. Therefore a particular member of the one dimensional family of parameters describing a particular curve must be chosen. The choice $\beta_0 = 1$ is the most convenient to work with (it is easy to implement within the framework of the ODR package) but it may result in numerical instability if β_0 is small compared to the remaining parameters. The above choice of coordinates, in which the curve is deliberately off set from the origin along the x axis should avoid this problem.

Therefore, in our fitting procedure, we fix $\beta_0 = 1$ and choose the remaining parameters $\beta_1, \ldots, \beta_5$ to minimize $d(\mathcal{C}_{\boldsymbol{\beta}}, \mathcal{S}')$ using the orthogonal distance regression package.

4.4.2 Initial Guess

To fit (4.13) using the ODR algorithm [21–23] (called ODRPACK) required a number of supporting functions and a first-guess of $\boldsymbol{\beta}$. The approach adopted here to computing the initial guess is to first select 14 points $(x_1, y_1), \ldots, (x_{14}, y_{14})$ along the selected branches of the trace. Ten of these points are evenly spaced along the length, with an extra two packed evenly into the two end intervals. These extra at the ends are to ensure that the initial guess pays particular attention to the ends of the trace. Substituting these 14 points into (4.13) gives 14 equations in five unknowns, which are solved in the least squares sense using the Moore-Penrose inverse to find an initial estimate of $\boldsymbol{\beta}$.

4.4.3 Pixel Weighting

The robustness of the fit can be improved by weighting each pixel in the fit according to its magnitude. In this way, an intense pixel is more likely to be closely fitted by the model. This can be done by rewriting (4.9) as:

$$\min_{\boldsymbol{\beta}} \quad \sum_{i=1}^{n} \|w_i \boldsymbol{\gamma}_i\|^2$$

$$\text{subject to} \quad f(\mathbf{x}_i + \boldsymbol{\gamma}_i : \boldsymbol{\beta}) = 0$$

where w_i are a set of weights that are proportional to the intensity of the i-th pixel in $\mathcal{S}$, the set of pixels in the thinned-trimmed ionogram trace.

4.4.4 Measuring Performance of Fitting Routines

Determining whether the model in (4.13) will satisfy the perturbation criteria in general is a difficult task because a precise mathematical description of all the possible traces is not available, or is too unwieldy. However, we can obtain an indication of the model measured against the perturbation criteria by examining the results of a number of the fits. In particular, the inverse condition number of each of the fits is a good indication of its quality — a value of one is ideal.

A scatterplot of the inverse condition numbers versus the length of the traces for the fits to the 4 test ionograms can be seen in figure 4.6. The inverse condition number of the ODR fits [23] is sometimes very small, indicating that the performance of the model in (4.13) is sometimes poor when measured against the perturbation criteria. We will examine the problematic traces in more detail in the next section.

4.5 Results

The results of fitting using the model in (4.13) can be seen in figures 4.15–4.26. Figures 4.15–4.18 depict the fit of the model to the original trimmed ionograms, and note the mess in figure 4.16 that is the result of fitting to the cluttered trace structure. Figures 4.23–4.26 show the dramatically improved fit that results when the improved trimming algorithm is used.

Note that five parameters describe each curve on the ionogram and that each curve has only been plotted in the region where the trace it fits resided. These four figures demonstrate that the feature extraction process presented here has done a good job of finding a small number of parameters that describe well the individual traces in most cases. The only exceptions are evident in figure 4.24. Nevertheless it is clear that on the simple ionograms the feature extraction process performs well.

Let us look more closely at the reasons for the poor fit of traces in test ionogram 2 (figure 4.24). Each trace has been individually numbered in the trimmed (by the improved algorithm) version of the ionogram as seen in figure 4.7. From figure 4.24 it can be seen that traces 6, 13, 14, 17, 20, 24, 27, and 29 have problems. The improved trimmed versions of these traces, along with the initial guesses and the final fits for each of them, can be seen enlarged in figure 4.8. We will now examine these traces and determine why the fitting routine has failed in each case.

Table 4.1 presents a number of details of the fitting procedure for each of the problematic traces of ionogram 2. Firstly note that shorter traces will be numerically less well conditioned than longer traces for the following reasons (see also figure 4.6). The shorter a trace segment, the more likely it is that it will look like a straight line. Furthermore, it is possible for smaller and smaller portions of an ever

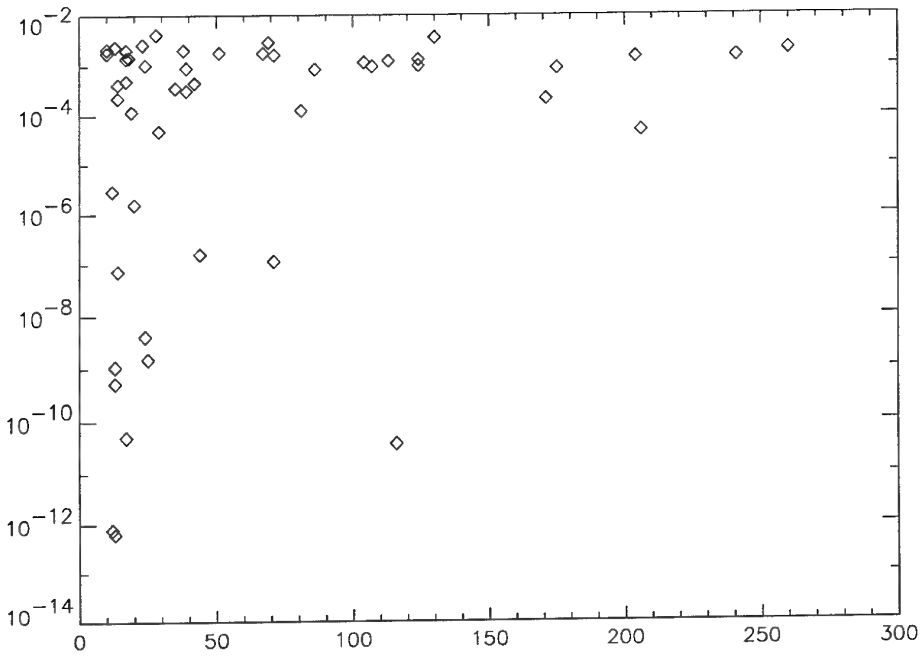


Figure 4.6: Scatterplot of log of the inverse condition numbers versus the length of the traces for the fits (figures 4.15–4.18) to the 4 test ionograms.

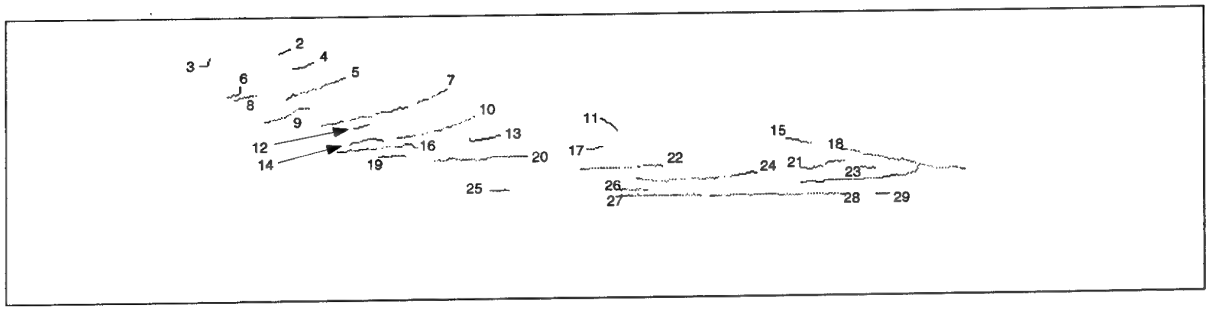


Figure 4.7: Ionogram 2, the version trimmed using the improved algorithm, with traces labelled.

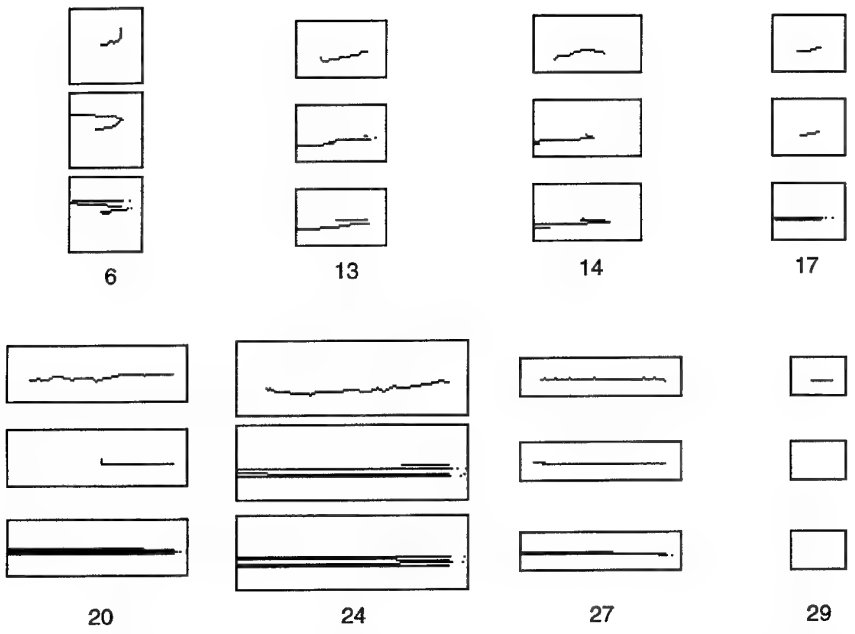


Figure 4.8: Traces 6, 13, 14, 17, 20, 24, 27, and 29 of ionogram 2, showing the trimmed versions (by the improved algorithm) of the trace first in each case, with the initial guess below and the fitted model shown at the bottom.

increasing model to look more and more like a straight line. This magnified region of the model thus fits the short trace segment ever more closely as the model parameters grow without bound, and so the fitting routine will never converge. In other words, the model parameters are in a region outside the basis of attraction of a finite-valued minimum and is in a region where the minimum is at infinity. It is important therefore to have a good initial guess for the model parameters to avoid this regime. The fitting failures of ionogram 2 are due to this problem, in addition to inadequacies of the model, fitting routine and plotting function as we will see.

Trace 6 is a rather short trace of 17 pixels, so this will have a negative impact upon the stability of the optimal fit. The initial guess for this trace has rather large values and a wild bend at the upper right termination of the trace, and this has thrown out the final fit (figure 4.8). In fact, the ODRPACK fitting algorithm did not converge before the 100 iteration limit was reached (table 4.1). This indicates that the initial guess computed by the methods outlined in subsection 4.4.2 is inadequate in this case. In addition, these problems have probably been exacerbated by weak performance of the ODRPACK fitting routine, as is noted in table 4.2. Suggested replacements for the initial condition computations and curve fitting algorithm are discussed in the next section.

Trace 13 has similar problems to those of trace 6, but with the distinction that its greater length has resulted in better numerical conditioning: the fitting algorithm converged on a parameter vector whose values are better scaled (after fitting, the largest β_i is not as large as for trace 6) and the inverse condition number is therefore an order of magnitude larger. There is evidence of unwanted bends at the ends of the initial guess which the fitting routine has not straightened out. The improvements required to fix the problems with this trace are consequently the same as for trace 13.

Trace 14 violates the assumption that the trace is largely a function x of y because it is downward bending at both ends. This has led to wild oscillations in the initial guess and large coefficient magnitudes. Furthermore, the fitting routine did not converge on a solution and was terminated at 100 steps. The coefficients increased in size in the final fit, making the oscillations even wilder. This behaviour highlights a weakness of trace model (4.13) for this trace.

Trace 17 is a very short trace, but the initial guess for it seems to be a good one in this case, because it is reasonable well scaled and has no evidence of bends at its ends. However, the fitting routine did not converge in 100 iterations and the final fit has wild oscillations. This is due to the fitting routine "wandering off" as it tries to strike the pixels ever more closely with the model by wildly oscillating

Table 4.1: Fitting Results for Problematic Traces of Ionogram 2

Trace Number	Length	Converged	Initial Guess			Final β			Inverse Cond. No.
			β_5	largest $\lvert\beta_i\rvert$	value	β_5	largest $\lvert\beta_i\rvert$	value	
6	17	no	-1.2	3	-123.9	-7.8	3	4338.1	4.94e-4
13	28	yes	-0.3	4	626.8	-0.24	4	557.0	0.0041
14	29	no	-0.4	3	571.3	-0.6	3	2987.7	4.8e-5
17	14	no	-2.0	2	40.9	0.3	2	570.9	7.3e-8
20	81	yes	8.4e-16*	4	416.7	8.3e-4	4	449.3	1.2e-4
24	104	yes	-0.0056	3	-72.3	0.0032	3	-80.6	0.0011
27	71	no	0.056	2	37.057	0.3	3	13556.3	1.1e-7
29	12	no	1.1e-16*	–	0	0.5	2	205.9	7.7e-13

it backwards and forwards across them. This problem can be fixed by replacing the fitting routine as discussed in the next section. I expect this is all that is required to correct the problem that this trace exhibits and so changing the model is not required here.

Trace 20 is practically horizontal and consequently the initial guess has set $\beta_5 \sim 0$, the coefficient of x. However, the initial guess has set some of the coefficients of y, in particular β_4, to be very large, and this should be discouraged in the initial guess. A new problem is evident here: there is a problem with the plotting of traces with $\beta_5 \sim 0$ and this has caused the strange right-angle bend at the left-hand end of the initial guess. This problem with plotting is not fatal, but it would be nice to have decent plots to examine in all cases and eliminate one possible source of confusion. However, writing a general plotting routine for implicit algebraic curves is not a trivial task, and I considered that my time was better spent in other areas. For this reason I have lived with these limitations. On this trace the fitting routine did converge on a solution, but it is a bad one with large coefficients and wild oscillations as the fitting routine has tried to "paint over" the locations of the pixels. These problems will be fixed using a better initial guess and fitting routine as explained in the next section.

There is another subtlety with this trace, however. Note that because $\beta_5 \sim 0$ in the initial guess, ODRPACK would normally eliminate this variable from consideration by treating it as fixed at zero for reasons that I will not go into here (see [23]). As a consequence, to overcome its treatment as a constant, I reset β_5 to the small but nonzero value of 0.001 whenever this occurs in the initial guess. This really is what is usually called a "hack" and probably contributes to my obtaining a final fit with wild oscillations. It should be corrected by using a better fitting routine to the one provided for implicit models in ODRPACK.

Trace 24 is suffering from the same problems as trace 14 only more so because the model is even less appropriate to the trace here than it is in the case of 14. It is not appropriate to this trace because the bending up of the trace at both ends means that the assumption that x is a function of y, or y is constant, has been violated. This means that the model has not been able to produce a good fit by a smooth curve, so it has painted over the pixels with wild oscillations. However, trace 24 has a better conditioned initial guess and final fit than trace 14, and it converged when 14 didn't, but this is no comfort and is simply due to its increased length.

Trace 27, similar to the case of trace 17, has a good initial guess, but the fitting routine did not converge on a solution and produced a fit with fairly wild oscillations, extremely large coefficients, and very poor conditioning at the solution. The remedy in this case is the same as trace 17: replace the fitting routine.

Trace 29 has no y variation in the pixels at all. As a consequence, all the coefficients of the initial guess are zero within the limit of precision. This should be tested for as a special case in the initial guess routine, and when it occurs the initial guess should be set to $y = c$, where c is a constant, and then the fitting routine can be skipped altogether. This is the best and simplest way to deal with this situation.

4.6 Discussion

We will now look into aspects of fitting, and consider ways of addressing the problems as suggested in table 4.2.

Table 4.2: Correcting the Fitting Problems of Traces in Ionogram 2

Trace Number	Problem fixed by improved:			
	Initial Guess	Plotting	Fitting Routine	Model
6	√		√	
13	√		√	
14				√
17			√	
20	√	√	√	
24				√
27			√	
29	√			

4.6.1 Advantages of Model Fitting Approach

Besides the advantages for merging and tracking, model fitting has other possibilities that could be exploited. Two obvious examples are in the detection of travelling disturbances and nose extensions.

Travelling disturbances in the ionosphere produce characteristic perturbations in the shape of ionogram traces in the nose region to form a "double nose" (figure 4.9(a)). Detecting their presence in an ionogram is of interest to ionospheric physicists. The model fit to traces could be used to produce a feature vector for the detection of these travelling disturbance artifacts (figure 4.9).

Nose extensions in ionogram traces (figure 4.10) are believed to be due to the presence of off great-circle reflection between the transmitter and receiver, and their presence in an ionogram is also of interest to ionospheric physicists. Their presence can also be detected using model fitting. This could be achieved by finding a roughly horizontal branch that extends out of the thinned-trimmed ionogram trace in the region where the bend of the trace fit occurs. This is demonstrated in figure 4.10.

4.6.2 Improved Fitting Algorithm

The ODRPACK [23] solution to the ODR problem in the case of an implicit model using the classic quadratic penalty function method to solve the constraint problem. Following the notation of (4.9), the penalty function used is

$$P(\boldsymbol{\beta}, \boldsymbol{\gamma}; r_k) = \sum_{i=1}^{n} \left(r_k \left[f(\mathbf{x}_i + \boldsymbol{\gamma}_i; \boldsymbol{\beta}) \right]^2 + \boldsymbol{\gamma}^{\mathrm{T}} \boldsymbol{\gamma} \right)$$

where r_k is the penalty term. ODRPACK solves a sequence of unconstrained minimization problems

$$\min_{\boldsymbol{\beta}, \boldsymbol{\gamma}} P(\boldsymbol{\beta}, \boldsymbol{\gamma}; r_k)$$

for a series of values of r_k tending to infinity. As a consequence of this, the speed of fitting is poor and the final fit suffers from all the false minima and quirks penalty function methods engender.

There are a number of other possible approaches to the fitting problem. These include various techniques to approximate the orthogonal distance of points from the curve [24–26]. The advice of Professor Powell [27,28] was to try the simplest and most direct approach: use a non-gradient based minimization procedure with an objective function that computes the sum of orthogonal distances. This objective function will be a rather complicated one, because it will have to compute the distance of each point from the curve using an optimization procedure. However, the closest point will not change much between each call to the objective function, so this could be made quite fast.

A very promising algorithm for fitting is presented by [29] that is essentially a Gauss-Newton algorithm with line search. In the region of the solution, the algorithm switches to a second order method

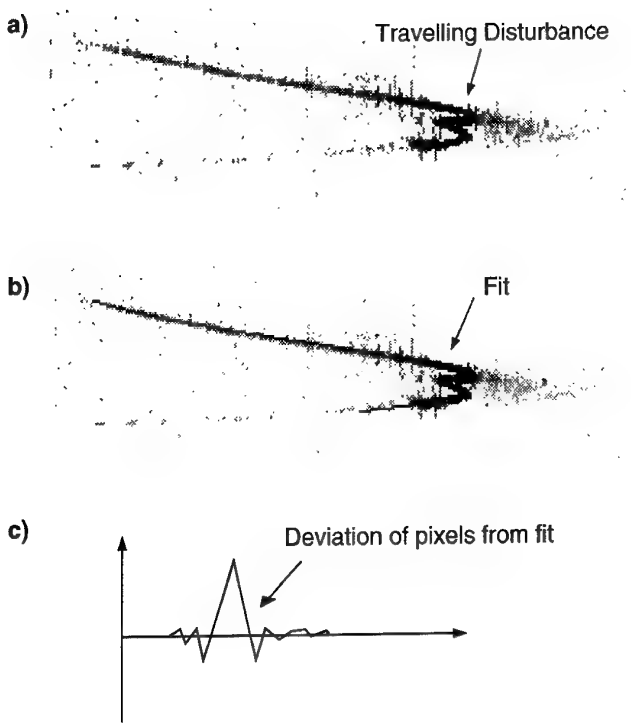


Figure 4.9: One application of the model fitting approach is to use the model's fit to obtain a feature vector based on the deviations of pixels in a trace's skeleton from the fit for use in the detection of travelling disturbances.

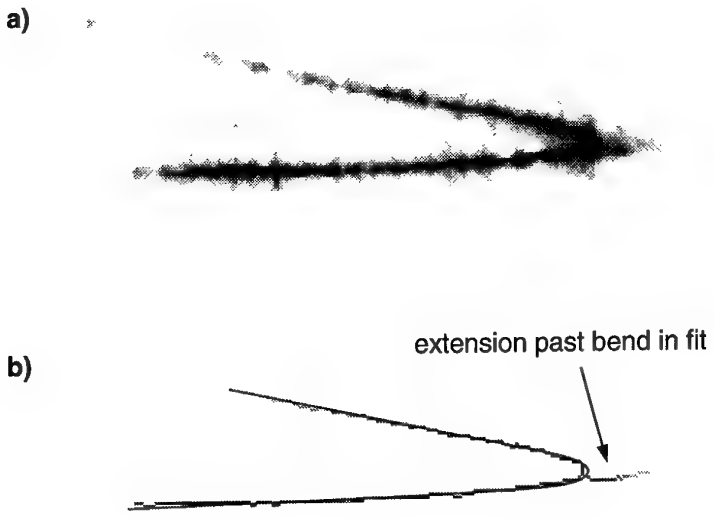


Figure 4.10: Nose extension could be detected using the model fitting approach.

to get quadratic convergence if progress towards the minimum is slow. Let $\tilde{\mathbf{x}}_i = \mathbf{x}_i + \boldsymbol{\gamma}_i$ be the closest point on the curve to $\mathbf{x}$ as in (4.9). Then formulation of the problem can be stated as

$$\min_{\boldsymbol{\gamma}_1,\ldots,\boldsymbol{\gamma}_n,\boldsymbol{\beta}} \quad \frac{1}{2}\sum_{i=1}^{n} \left\| \mathbf{W}_i^{1/2}(\tilde{\mathbf{x}}_i - \mathbf{x}_i) \right\|^2$$

$$\text{subject to} \quad f(\tilde{\mathbf{x}}_i;\boldsymbol{\beta}) = 0, \quad i = 1,\ldots,n$$

where $\mathbf{W}$ is a matrix of non-negative weights for every element of the residual. The authors linearize the constraints to obtain

$$\min_{\delta\boldsymbol{\gamma}_i,\delta\boldsymbol{\beta}} \quad \frac{1}{2}\sum_{i=1}^{n} \left\| \mathbf{W}_i^{1/2}(\tilde{\mathbf{x}}_i + \delta\tilde{\mathbf{x}}_i - \mathbf{x}_i) \right\|^2$$

$$\text{subject to} \quad \mathbf{E}\delta\tilde{\mathbf{x}} + \mathbf{F}\delta\boldsymbol{\beta} = -\mathbf{f}, \quad i = 1,\ldots,n$$

where $\mathbf{E}$ and $\mathbf{F}$ are matrices of second derivatives of f with respect to $\tilde{\mathbf{x}}$ and $\boldsymbol{\beta}$, respectively. They then eliminate the constraints using a weighted QR factorization to give an unconstrained optimization problem. This approach is considerably more sophisticated than the penalty function method used in ODRPACK, and it allows for infinite and zero weights in $\mathbf{W}$. Using an algorithm such as this would give a great improvement in the fits over that obtained by ODRPACK and should address all the fitting routine problems noted in table 4.2.

4.6.3 Improved Initial Guess

The method presented in subsection 4.4.2 for determining an initial guess at the parameter values makes use of only 14 of the set of pixels comprising a thinned-trimmed ionogram trace. We can improve on this at little computational cost so that all the points in this set are used for the initial guess. This is done in the following way.

As in (4.3), let our model function be denoted by

$$f(\mathbf{x},\boldsymbol{\beta}) = \sum_{i=0}^{P}\sum_{j=0}^{Q} \beta_{ij} x^i y^j.$$

The "proper" approach to finding an initial guess for $\boldsymbol{\beta}$, the vector of parameters β_{ij}, would be to find a reasonable solution to the problem given by

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \sum_{k=1}^{n} \gamma_k^2 + \epsilon_k^2 \\ \text{subject to} \quad & \sum_{i=0}^{P}\sum_{j=0}^{Q} \beta_{ij}(x_k+\gamma_k)^i(y_k+\epsilon_k)^j = 0, \quad k = 1,\ldots,n \\ & \|\boldsymbol{\beta}\| = 1 \end{aligned} \tag{4.14}$$

which is equivalent to the fitting problem in (4.9) with an additional constraint on $\boldsymbol{\beta}$. This additional constraint is to ensure that the fitting problem is better normalized and so eliminate the problems of large coefficients noted in section 4.5. Of course, solving (4.14) for an initial guess is out of the question, because to solve it would be to solve the whole fitting problem.

A "quick and dirty" solution to (4.14) is given by the solution to

$$\min_{\substack{\boldsymbol{\beta} \\ \|\boldsymbol{\beta}\|=1}} \quad \sum_{k=1}^{n} \left| \sum_{i=0}^{P}\sum_{j=0}^{Q} \beta_{ij} x_k^i y_k^j \right|^2. \tag{4.15}$$

By defining the matrix $\mathbf{A}$ to be

$$\mathbf{A}_{k,ij} = x_k^i y_k^j$$

(using lexicographic ordering of the second index ij) then (4.15) can be rewritten as

$$\min_{\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\beta}=1} \boldsymbol{\beta}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\boldsymbol{\beta}. \tag{4.16}$$

It is well known that the solution to (4.16) is given by $\boldsymbol{\beta} = \mathbf{v}_{\min}$, where $\mathbf{v}_{\min}$ is the right singular vector corresponding to the smallest singular value of $\mathbf{A}$, denoted by $\sigma_{\min}$. In this case,

$$\boldsymbol{\beta}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\boldsymbol{\beta} = \mathbf{v}_{\min}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{v}_{\min} = \sigma_{\min}^2.$$

The problem with the "quick and dirty" approach is that the residual

$$\left| \sum_{i=0}^{P} \sum_{j=0}^{Q} \beta_{ij} x_k^i y_k^j \right|^2$$

has no geometric interpretation. That is, a rescaling of the problem in (4.16) can produce an entirely different minimum $\boldsymbol{\beta}$ which will be wildly different in how well it solves the problem in (4.14). We can alleviate this normalization problem by down-weighting each row in $\mathbf{A}$ by the size of vector $\mathbf{A}_k$ and form the new matrix $\mathbf{B}$ with elements given by

$$\mathbf{B}_{kl} = w_k \mathbf{A}_{kl}$$

where

$$w_k = \frac{1}{[\sum_l \mathbf{A}_{kl}^2]^{1/2}}.$$

The problem now becomes

$$\min_{\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\beta}=1} \sum_{k=1}^{n} w_k \left| \sum_l \beta_l \mathbf{A}_{kl} \right|^2 \iff \min_{\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\beta}=1} \boldsymbol{\beta}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{B}\boldsymbol{\beta}. \tag{4.17}$$

In conclusion, the solution to (4.17) employs all the pixels in the trace to compute an initial $\boldsymbol{\beta}$ that is normalized such that $\|\boldsymbol{\beta}\| = 1$. This will overcome the problems of large coefficients encountered in trace 20 and the wild oscillations in traces 6 and 13. Note that with this approach, the constant term is no longer set to one (section 4.4.1) and so β_0 has a non-zero value. Therefore, in the fitting procedure, this must be treated as a fixed weight to avoid a zero-valued solution.

4.6.4 Improved Model

Traces 14 and 24 of test ionogram 2 have highlighted an inadequacy of the existing model in (4.13) (see table 4.2). These traces violated the assumption that all traces are either a function y of x, or y is constant, and they show the need to cater for traces that bend at both ends. To achieve this we need to add an x^2 term to the model. Thus, the trace model is the family of curves defined by

$$\mathcal{C}_{\boldsymbol{\beta}} \equiv \{(x,y) : f(\mathbf{x} : \boldsymbol{\beta}) = \beta_0 + \sum_{i=1}^{4} \beta_i y^i + \beta_5 x + \beta_6 x^2 = 0\}. \tag{4.18}$$

Note that in (4.18) above the β_0 is not normalized to one, but it will need to be fixed throughout the fitting procedure at the value determined by the initial guess routine (of the previous subsection) so that the trivial zero solution is avoided.

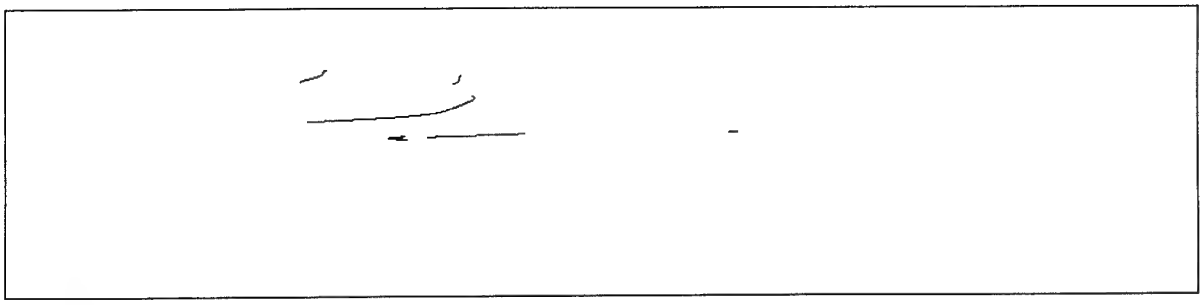


Figure 4.11: Ionogram 1 — initial guess for implicit multivariate quartic polynomial fit.

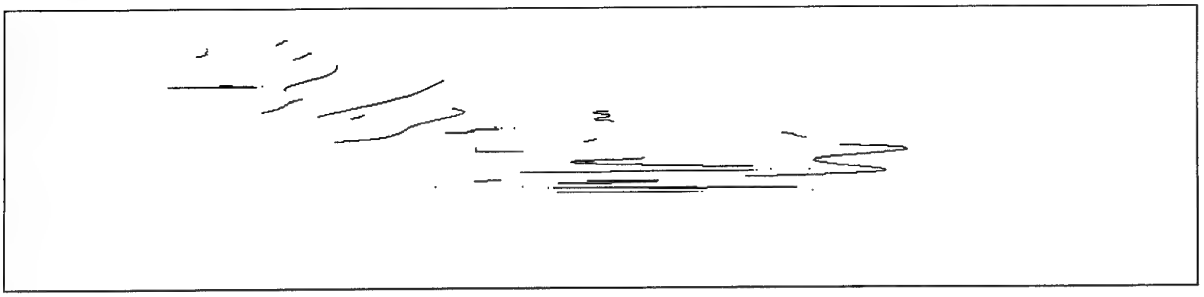


Figure 4.12: Ionogram 2 — initial guess for implicit multivariate quartic polynomial fit.

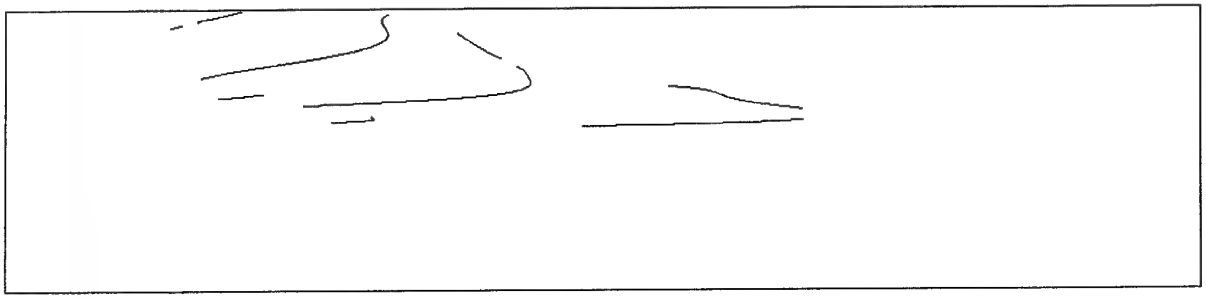


Figure 4.13: Ionogram 3 — initial guess for implicit multivariate quartic polynomial fit.

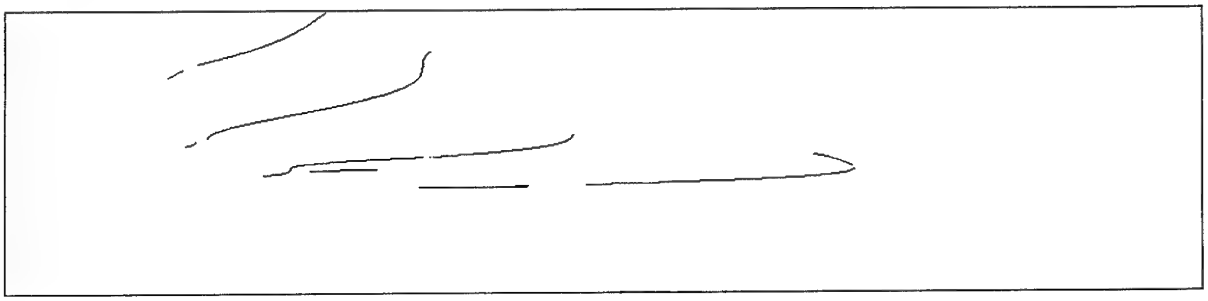


Figure 4.14: Ionogram 4 — initial guess for implicit multivariate quartic polynomial fit.

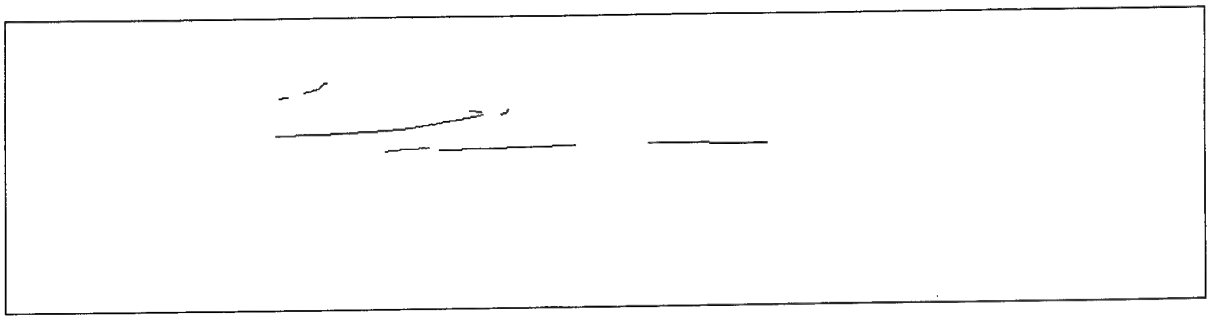


Figure 4.15: Ionogram 1 — implicit multivariate quartic polynomial fit.

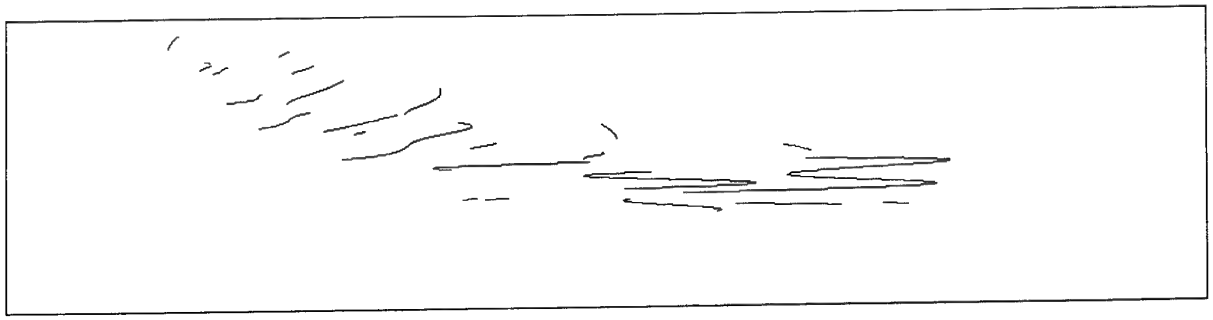


Figure 4.16: Ionogram 2 — implicit multivariate quartic polynomial fit.

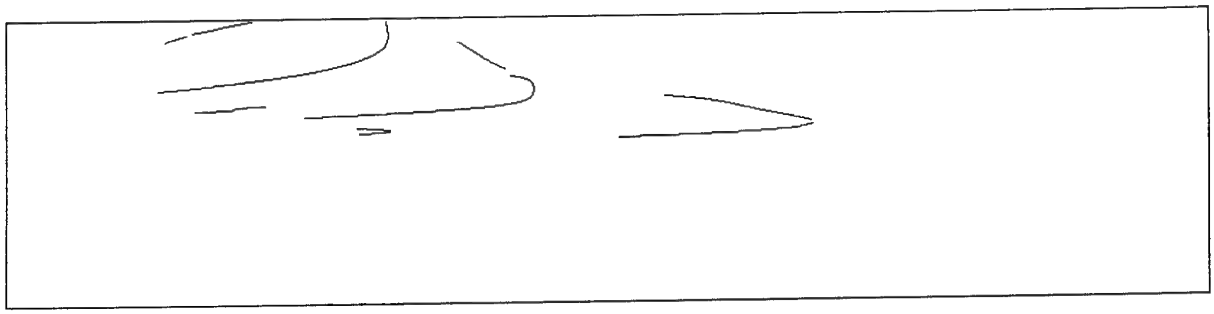


Figure 4.17: Ionogram 3 — implicit multivariate quartic polynomial fit.

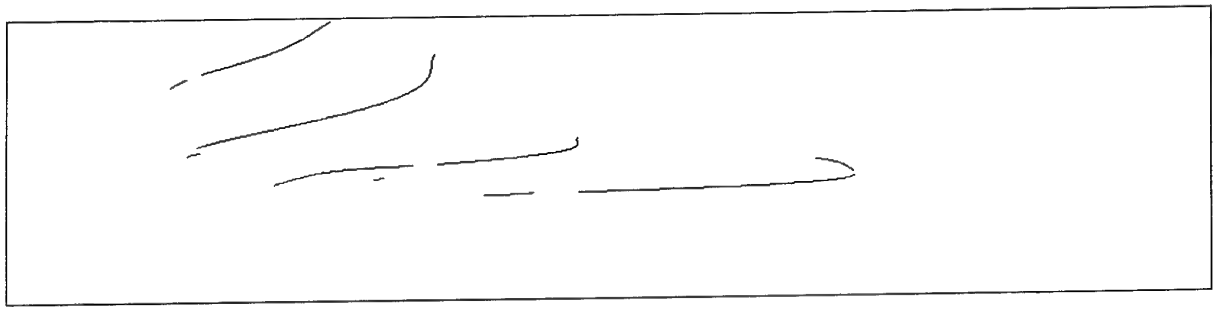


Figure 4.18: Ionogram 4 — implicit multivariate quartic polynomial fit.

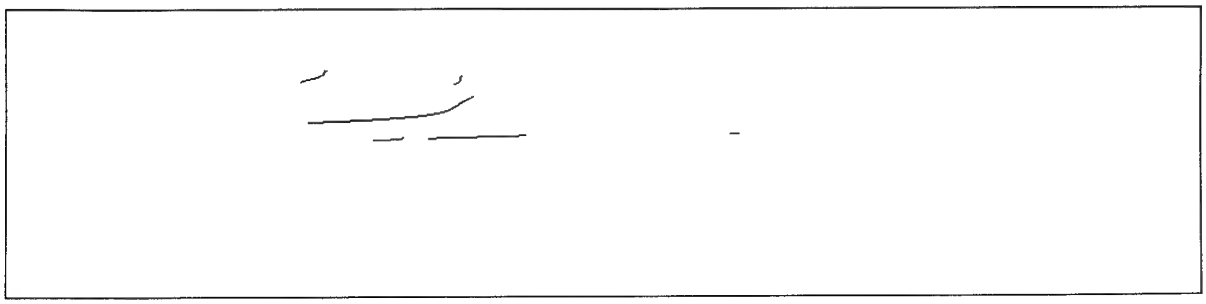


Figure 4.19: Ionogram 1 — initial guess for implicit multivariate quartic polynomial fit (hand trimmed).

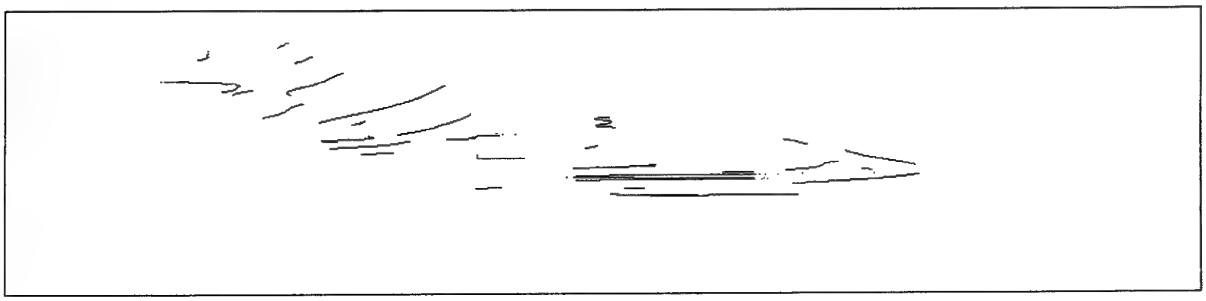


Figure 4.20: Ionogram 2 — initial guess for implicit multivariate quartic polynomial fit (hand trimmed).

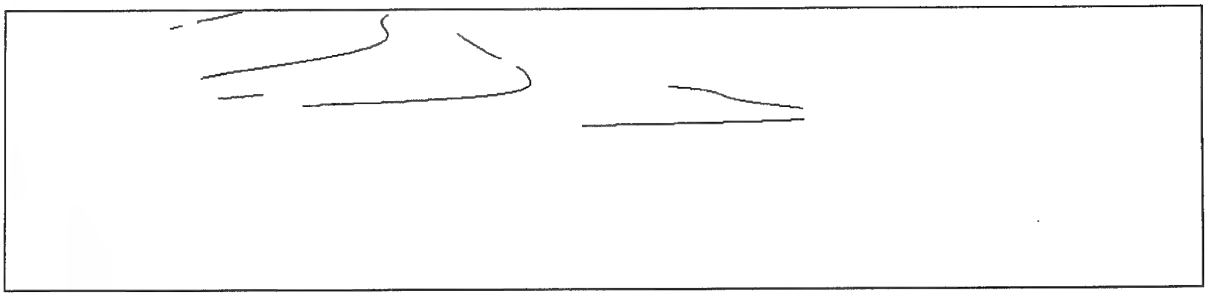


Figure 4.21: Ionogram 3 — initial guess for implicit multivariate quartic polynomial fit (hand trimmed).

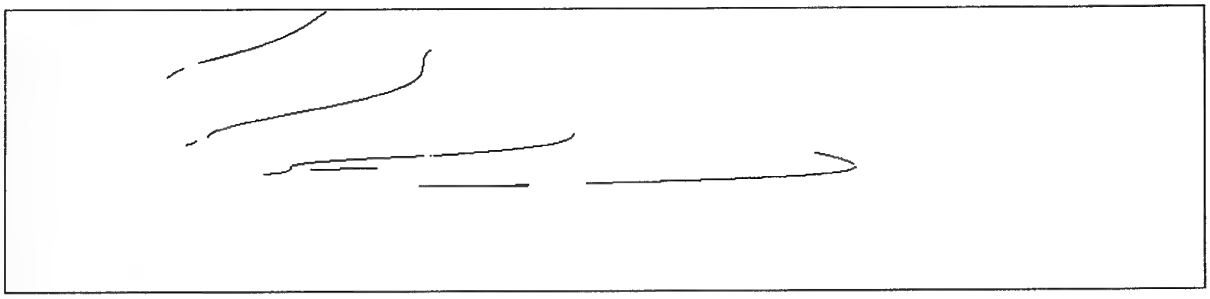


Figure 4.22: Ionogram 4 — initial guess for implicit multivariate quartic polynomial fit (hand trimmed).

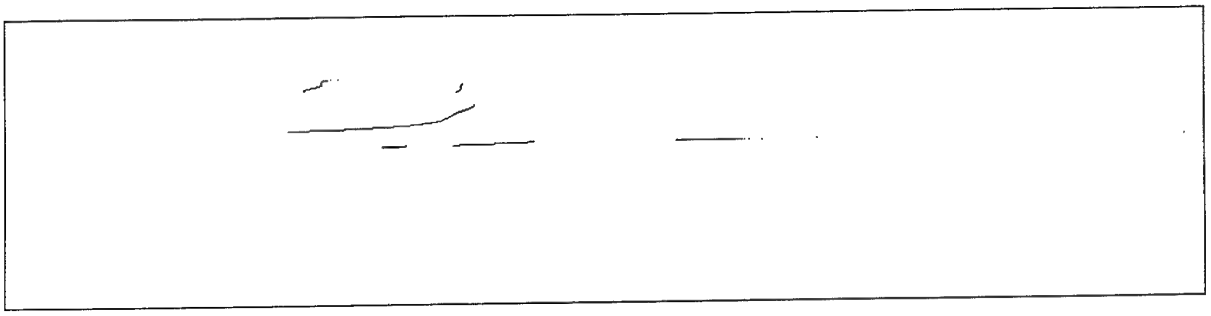


Figure 4.23: Ionogram 1 — implicit multivariate quartic fit to hand-trimmed traces.

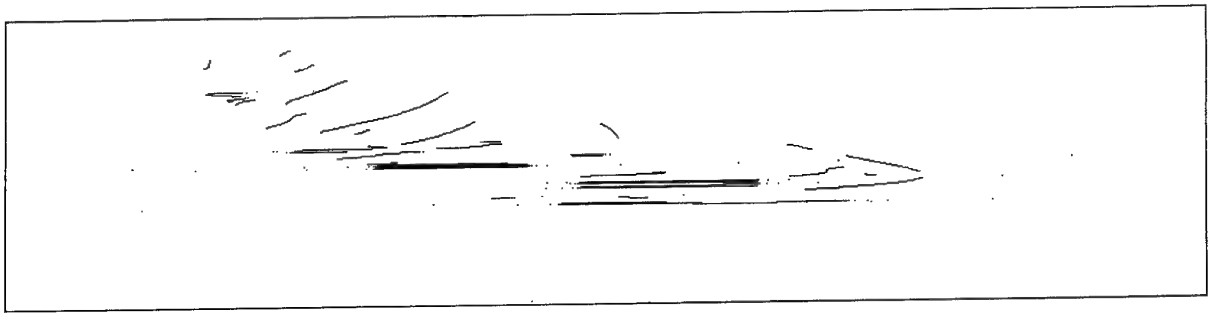


Figure 4.24: Ionogram 2 — implicit multivariate quartic fit to hand-trimmed traces.

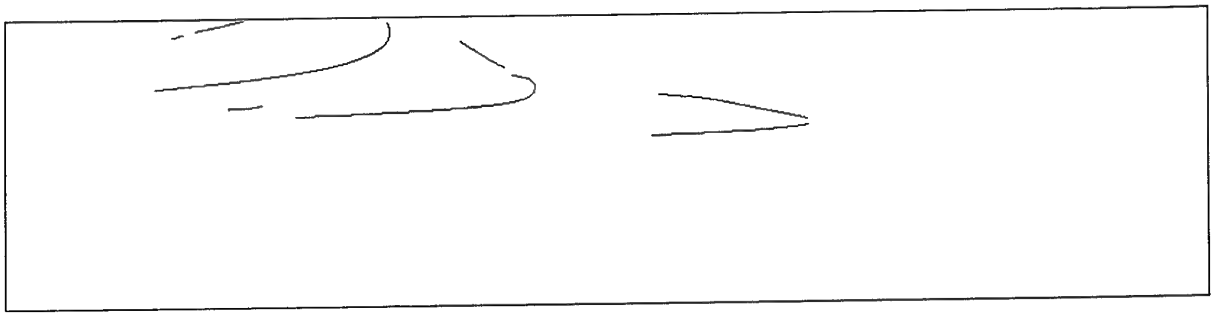


Figure 4.25: Ionogram 3 — implicit multivariate quartic fit to hand-trimmed traces.

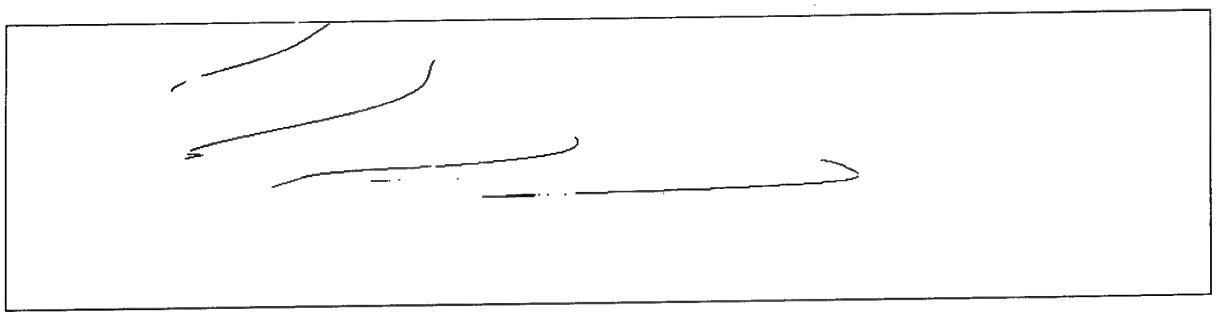


Figure 4.26: Ionogram 4 — implicit multivariate quartic fit to hand-trimmed traces.

Chapter 5

Feature Extraction: Merging

This chapter details a further aspect of the feature extraction process that is designed to account for broken or missing features in an ionogram. *Merging*, is the process of reconnecting trace fragments, which clearly belong to the same propagation mode, that have become disconnected for some reason (noise, attenuation, *etc.*) and so have been fitted separately. Thus merging is one more step designed to move towards the ideal where one trace corresponds to one propagation mode and vice versa.

Figure 5.1 depicts a situation where two traces, which are clearly due to the same propagation mode, have produced separate trace fits, and consequently, each has a different collection of parameters that describes its shape. Obviously, we would like to be able to represent these two traces with the one feature vector for the sake of economy. Not only that, we would like to know that they are really connected (in the sense of being produced by the same propagation mode) so that when we come to mode identification the ideal that each trace has a single, different and distinct propagation mode associated with it will be true. Through the improved trimming algorithm of section 3.6, we have already ensured that each trace fragment has a single generating propagation mode. Now we are ensuring that for each propagation mode there is only one trace that it generated.

There are many candidates for merging. Any group of two (or more) trace fits could possibly be due to the one propagation mode and consequently are candidates. Furthermore, merging is a difficult task, and I have found that no single test for merging works in all cases. Therefore a battery of tests is required to solve this problem. In this chapter I will present a range of tests that vary in sophistication to apply to the merge problem.

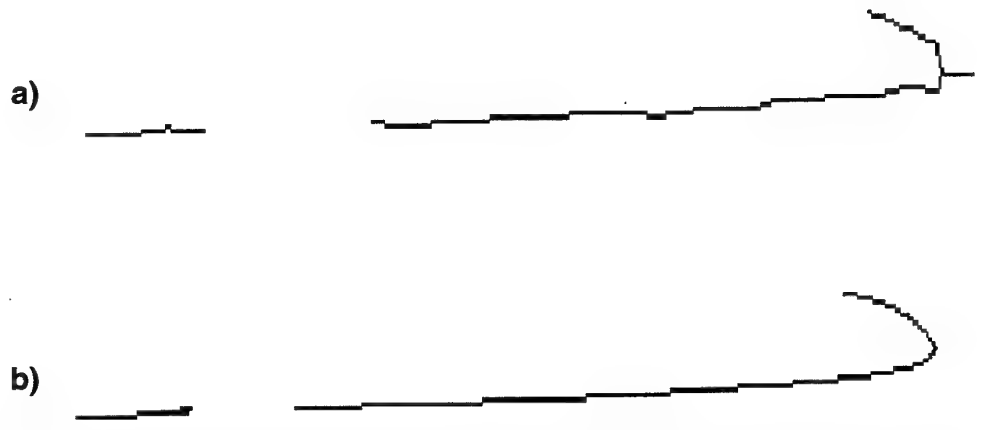


Figure 5.1: Two thinned trace segments that are clearly due to the same propagation mode are depicted in (a) and their corresponding fits in (b).

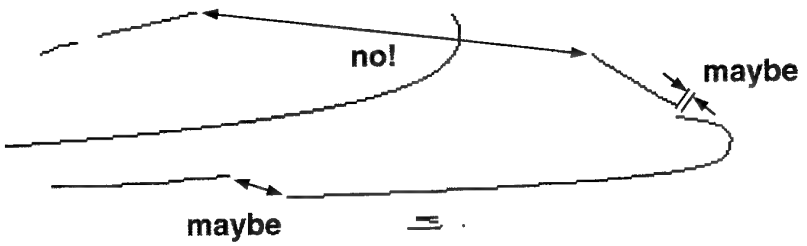


Figure 5.2: The first test for merging examines the distance between the closest end-points of two traces. If it is less than a threshold, then the two traces could possibly be merged.

5.1 Merge Tests

The obvious brute force approach to merging involves the following steps. Firstly, we would find the error in the best fitting curves to each segment independently, and we can do this simply by recording it during the fitting procedure. Secondly, we must find the best fitting curve to both segments simultaneously and then compare the error in the joint fit with the errors in the two independent fits and decide whether the joint fit is adequate.

The problem with this approach is that the computational cost of repeatedly fitting curves is very expensive, especially if the approach is extended to tackle the problem of deciding whether three of more segments are all part of the one trace. This combinatorial explosion of joint fits means that this approach is just not feasible. Because of this high cost, cheaper alternatives are necessary. We will now consider a number of such tests for merging.

5.1.1 Test 1: Distance Between End-Points

The first test of merging is also conceptually and computationally the simplest. It involves determining if the distance between the closest end-points of two candidate traces is less than a threshold. This threshold is set conservatively large so that all traces that should be merged will be closer, but at the same time many traces that should not be merged are eliminated as candidates. This is depicted in figure 5.2.

5.1.2 Test 2: Overlap in Group Delay

The second test for merging, called the *overlap in group delay* test, tries to detect when a trace completely overlaps another along the y-axis. If the traces are separated in the range of their y values then all is well — they could possibly be merged. If not, and one trace completely overlaps the other, then the traces cannot be merged. A small margin (of 5 pixels) is ignored for the purposes of computing overlap around the edges of the enveloping trace. This is to ensure that horizontal traces, which will overlap in group delay, will still be considered as candidates. Lastly, if there is a partial overlap of the y values merging is still possible. These principles are depicted in figure 5.3.

5.1.3 Test 3: Midpoints

The third test compares distances to midpoints with distances between endpoints to try and determine if one trace is bending away from the other. More precisely, this test determines if there is a midpoint on one trace that is closer to the other trace than the minimum distance between their endpoints (less a margin of 5 pixels). The margin is to allow for small turnings away at the end of a trace due to noise. If the trace fragments do bend away from one another significantly, then it is not likely that they are due to the same propagation mode, and so they should not be merged. See figure 5.4 for an example of the application of these principles.

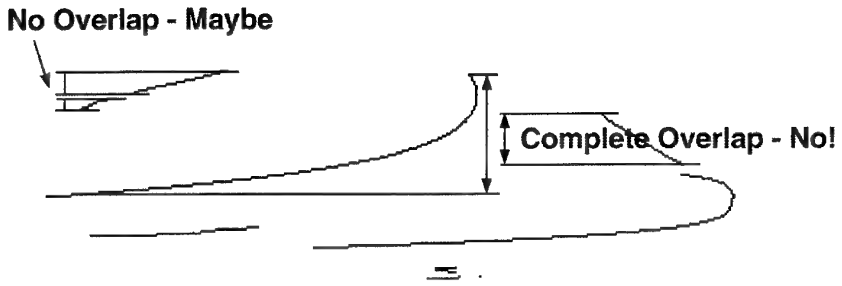


Figure 5.3: Test 2, the overlap in group delay test, seeks to identify when a trace completely overlaps another along the y axis. If do, then the two traces cannot be merged.

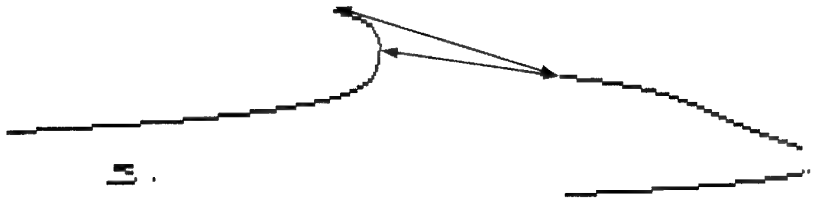


Figure 5.4: The third test for merging, the midpoints test, identifies when one trace bends away from another at its closest end by looking for closer midpoints along the trace. If a trace bends away, as these do, then the traces cannot be merged.

5.1.4 Test 4: Equidistance

This test extends on the principles behind the midpoints test. There we tested to see if two traces bent away from one another as an indication that merging them should not take place. Here we go one step further: if the traces are roughly parallel in a region of substantial overlap (along either the x or y axes) at their closest ends (the ends that would be joined if they could be), then they should not be merged. An example of this occurs in figure 5.5.

5.1.5 Test 5: Horizontals

This test, called the horizontals test, is designed to eliminate merge candidates where each trace is predominately horizontal and at significantly different y-values (figure 5.6). When such situations occur, it is obvious that the two traces cannot be part of a larger trace due to the same propagation mode.

We can test for these situations simply in the following manner. Firstly, we determine if the two traces are separated in their y-range — if they are not then terminate this test as nothing more can be

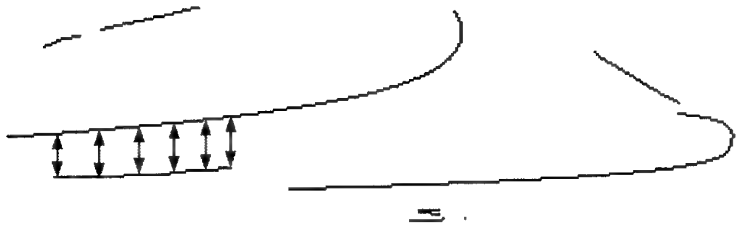


Figure 5.5: The fourth test, the equidistance test, identifies when two traces are roughly parallel in a region where they substantially overlap at their closest ends. If any such ends are roughly parallel, then they cannot be merged.

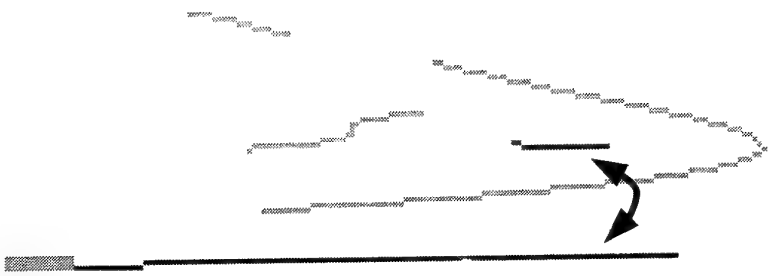


Figure 5.6: Two traces are indicated that would be eliminated from consideration as a possible merge by the horizontals test.

determined. If they are, then it is necessary to determine that their ends are either overlapping or not too far apart along the x axis — the closer they are, the sharper the bend needs to be to merge them, and the less desirable the merge becomes. A separation of less than the average of their two lengths projected along the x axis is deemed to be acceptable to continue with this test. A separation of more than this indicates once again that this test is not applicable and it cannot determine anything. Finally, we test to see if the y-range from the top of the upper trace to the bottom of the lower trace is more than two times the y-range of the individual traces. If it is, then the two traces cannot be merged. If not, then this test again cannot determine anything certain about the situation.

5.1.6 Common Coordinate Systems

In section 4.4.1 I showed how to transform to local coordinates so that fitting is reasonably well conditioned. As a result, in the merging problem, we will have two such local coordinate systems, one for each trace under consideration. The problem is that we need to have the two traces in the one common coordinate system for the remaining merge tests (figure 5.7).

Let $\mathcal{S}_1$ and $\mathcal{S}_2$ denote the set of points corresponding to trace 1 and 2, respectively. Further, let $\mathbf{x}_i$ and $\mathbf{x}'_i$ denote the image plane and transformed local coordinates, respectively, of trace i, with translation and scale parameters $\mathbf{u}_i$ and α_i given by

$$\mathbf{x}'_i = \alpha_i(\mathbf{x}_i - \mathbf{u}_i), \qquad i = 1, 2. \tag{5.1}$$

Note that, from (4.12), $\mathbf{u}_i$ is given by

$$\mathbf{u}_i = (u_i, v_i) = (\frac{x_{i_{\max}} + x_{i_{\min}}}{2}, y_{i_{\min}}), \qquad i = 1, 2$$

and that

$$\alpha_i = \frac{1}{y_{i_{\max}} - y_{i_{\min}}}, \qquad i = 1, 2.$$

We now need to determine a common coordinate system for $\mathcal{S}_1$ and $\mathcal{S}_2$ in which to carry out the merging test. We will denote this common coordinate system by $\mathbf{x}''$.

Just as explained in section 4.4.1 for fitting, for the remaining merge tests to be well conditioned in the common coordinate system, the mapping to the common coordinates must satisfy

$$[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \qquad \rightarrow \qquad [-z, z] \times [0, 1]$$

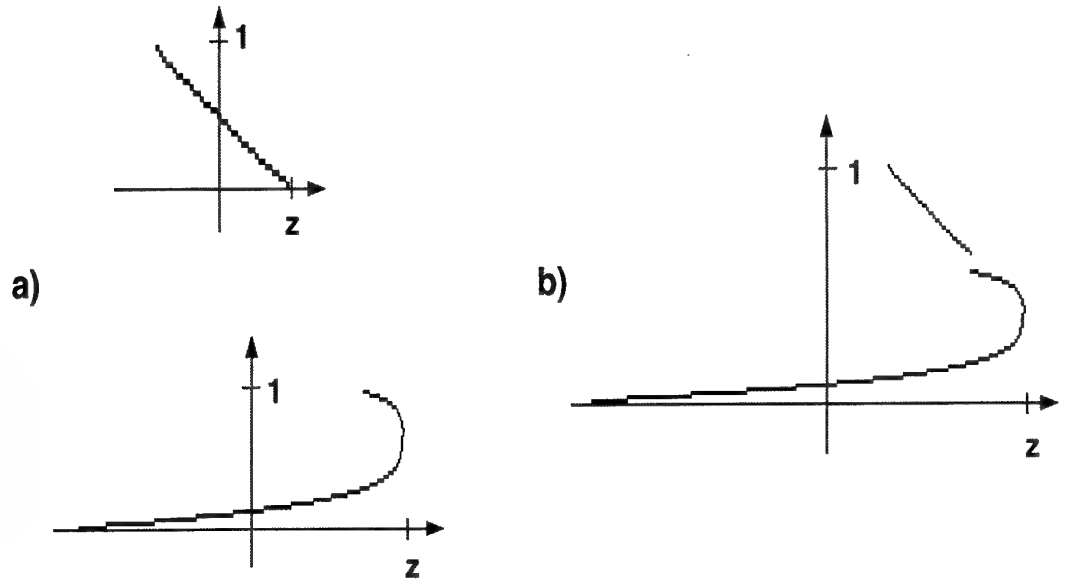


Figure 5.7: Two traces have been fitted in their own normalized coordinate system in (a). Before some merge tests can be undertaken, the two local coordinate systems have to transformed into one common one (b).

where

$$\begin{aligned} x_{\max} &= \max\{x_{1_{\max}}, x_{2_{\max}}\}, \\ x_{\min} &= \min\{x_{1_{\min}}, x_{2_{\min}}\}, \\ y_{\max} &= \max\{y_{1_{\max}}, y_{2_{\max}}\}, \\ y_{\min} &= \min\{y_{1_{\min}}, y_{2_{\min}}\}. \end{aligned}$$

This gives that

$$z = \frac{x_{\max} - x_{\min}}{2(y_{\max} - y_{\min})}$$

so that the mapping to the common coordinates $\mathbf{x}''$ is given by

$$\mathbf{x}'' = \alpha(\mathbf{x} - \mathbf{u}) = \frac{1}{y_{\max} - y_{\min}}(\mathbf{x} - (\frac{x_{\max} + x_{\min}}{2}, y_{\min})) \tag{5.2}$$

as before. It now remains to determine the transformation of each of the individual local trace coordinates to this new common coordinate system.

We know that traces 1 and 2 are locally transformed according to (5.1). Therefore, by substituting for the image coordinates in (5.2), we get that the transformation from the local image coordinates $\mathbf{x}'_i$ of trace i to the common coordinates $\mathbf{x}''$ is given by

$$\mathbf{x}'' = \frac{\alpha}{\alpha_i}(\mathbf{x}'_i - \alpha_i(\mathbf{u} - \mathbf{u}_i)).$$

Expressing this solely in terms of parameters in the two local coordinate systems we get that

$$\alpha = \frac{1}{\max_i\{\frac{1}{\alpha_i} + v_i\} - \min_i\{v_i\}}$$

and

$$\mathbf{u} = (\frac{1}{2}(\max\{\frac{\mathbf{x}'_{i_{\max}}}{\alpha_i} + u_i\} - \min\{\frac{\mathbf{x}'_{i_{\min}}}{\alpha_i} + u_i\}), \min\{v_i\}).$$

5.1.7 Test 6: Second Order Approximation

This next test is mathematically the most difficult test[1] and is also significantly novel [30]. It relies on the principle that the Hessian of the fit to each trace segment alone can be used to compute a second order approximation of the joint fit. This test proceeds in the following manner.

1. Find the best fitting curves to each segment independently (Chapter 4).
2. Compute the Hessians for these fits and use them to construct a second order approximation to the fitting metric as a function of the curve parameters.
3. Form the sum of second order approximations to fits to both traces in a common coordinate system and compute the parameters that minimize this approximation.
4. Decide whether the error in the overall approximation is significantly larger than the errors of the two independent best fits.

The advantages of this approach are firstly that it is very fast because once the initial independent fits have been computed only low order matrix manipulations are needed. Secondly, it can be easily extended to looking at joint fits of more than two traces. Its disadvantages stem from the fact that it is based on a second order approximation. This approximation may not be very accurate, especially in ill-conditioned problems that are the result of poor parameterizations or poorly behaved curves. Furthermore, some fitting metrics, in particular orthogonal distance regression (ODR), may have discontinuities in their second derivatives. This may happen in ODR when the curve has two branches and the data point is midway between them. A slight perturbation in parameters may now jump the closest point on the curve to the data point from one branch of the curve to the other in a discontinuous manner.

Experience has shown that this test is the most sensitive for determining whether two traces should be joined when the decision is a close one. However, when the traces are of wildly different placement, this test can actually be wrong, because the second order approximation to the joint fit is extremely poor. Therefore, I have set out to eliminate these wild cases firstly using tests 1–4.

We will next consider how to obtain an accurate second order approximation to the fitting metric for a single trace.

Constructing Second Order Approximations to the Fitting Metric

In the course of finding the best fitting curves $\mathcal{C}_{\boldsymbol{\beta}_1}$ and $\mathcal{C}_{\boldsymbol{\beta}_2}$ to two segments $\mathcal{S}_{\boldsymbol{\beta}_1}$ and $\mathcal{S}_{\boldsymbol{\beta}_2}$, the fitting routine will minimize the value (see (4.4))

$$d_i = \sum_{n=1}^{N_i} \min_{\mathbf{x} \in \mathcal{C}_{\boldsymbol{\beta}_i}} \|\mathbf{x}_{i,n} - \mathbf{x}\|^2$$

where d_i is the error of the best fit. Note that the parameter vector $\boldsymbol{\beta} = [\beta_0, \ldots, \beta_5]^{\mathrm{T}}$ for model function (4.13) has the element β_0 normalized to one (section 4.4.1) or some other fixed value when the improved initial guess algorithm is used (section 4.6.3). Let $\boldsymbol{\beta}_i^* = [\beta_1, \ldots, \beta_5]^{\mathrm{T}}$ be defined to be the vector of free parameter values as determined by the fitting routine, and $\mathbf{H}_i^*$ be the matrix of second derivatives showing the sensitivity of d_i to changes in $\boldsymbol{\beta}_i^*$.

Now because $\boldsymbol{\beta}_i^*$ was determined by fitting under the assumption that $\beta_0 = 1$ (or some other fixed value), it makes sense to consider the larger vectors

$$\boldsymbol{\beta}_i = \begin{bmatrix} 1 \\ \boldsymbol{\beta}_i^* \end{bmatrix} \tag{5.3}$$

and consider the sensitivity to d_i to changes with respect to $\boldsymbol{\beta}_i$. Let $\mathbf{H}_i$ be the Hessian for d_i with respect to the full vector $\boldsymbol{\beta}_i$, so that to a second order approximation

$$d_i(\boldsymbol{\beta}) \sim d_i + \frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\beta}_i)^{\mathrm{T}} \mathbf{H}_i (\boldsymbol{\beta} - \boldsymbol{\beta}_i) \tag{5.4}$$

[1] In fact, this particular test took longer to develop and get working than any other existing module in the entire autoscaling system.

(because the first derivative will be zero at the minimum determined by the fitting procedure). We can determine $\mathbf{H}_i$ from $\mathbf{H}_i^*$ and the condition that $d_i(\boldsymbol{\beta}) \equiv d_i(\boldsymbol{\beta}/\|\boldsymbol{\beta}\|)$ should be invariant to changes of magnitude in the best fitting parameters. To see this, first let

$$\mathbf{H}_i = \begin{bmatrix} n_i & \mathbf{h}_i^{\mathrm{T}} \\ \mathbf{h}_i & \mathbf{H}_i^* \end{bmatrix}, \tag{5.5}$$

where n_i and $\mathbf{h}_i^{\mathrm{T}}$ will be determined after the following development. Next, note that the condition $d_i(\boldsymbol{\beta}_i) = d_i(\alpha\boldsymbol{\beta}_i)$ for all α implies that

$$d_i(\alpha\boldsymbol{\beta}_i) \sim d_i + \frac{(1-\alpha)^2}{2}\boldsymbol{\beta}_i^{\mathrm{T}}\mathbf{H}_i\boldsymbol{\beta}_i = d_i(\boldsymbol{\beta}_i) = d_i.$$

Therefore $\boldsymbol{\beta}_i^{\mathrm{T}}\mathbf{H}_i\boldsymbol{\beta}_i = 0$ and $\mathbf{H}_i\boldsymbol{\beta}_i = 0$ since $\mathbf{H}_i$ is positive semi-definite as it is the matrix of second derivatives at a minimum. Therefore,

$$\begin{bmatrix} n_i & \mathbf{h}_i^{\mathrm{T}} \\ \mathbf{h}_i & \mathbf{H}_i^* \end{bmatrix} \begin{bmatrix} 1 \\ \boldsymbol{\beta}_i^* \end{bmatrix} = 0$$

and so

$$\begin{bmatrix} n_i \\ \mathbf{h}_i \end{bmatrix} = - \begin{bmatrix} \mathbf{h}_i^{\mathrm{T}}\boldsymbol{\beta}_i^* \\ \mathbf{H}_i^*\boldsymbol{\beta}_i^* \end{bmatrix}$$

which gives that the components $\mathbf{h}_i$ and n_i of $\mathbf{H}_i$ in (5.5) are given by

$$\begin{aligned} \mathbf{h}_i &= -\mathbf{H}_i^*\boldsymbol{\beta}_i^* \\ n_i &= \boldsymbol{\beta}_i^{*\mathrm{T}}\mathbf{H}_i^*\boldsymbol{\beta}_i^*. \end{aligned}$$

Note that the condition $\mathbf{H}_i\boldsymbol{\beta}_i = 0$ implies that (5.4) can be reduced to

$$d_i(\boldsymbol{\beta}) \sim d_i + \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}}\mathbf{H}_i\boldsymbol{\beta}.$$

However, this is not the most accurate approximation we can construct for $d_i(\boldsymbol{\beta})$ since it satisfies

$$d_i(\boldsymbol{\beta}) = d_i\left(\frac{\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|}\right) \tag{5.6}$$

only for $\boldsymbol{\beta} = \boldsymbol{\beta}_i$. If instead we were to construct an approximation of the form

$$d_i(\boldsymbol{\beta}) \sim d_i + \frac{1}{2}\frac{\boldsymbol{\beta}^{\mathrm{T}}\mathbf{L}_i\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^2}$$

then this would satisfy (5.6) for all $\boldsymbol{\beta}$. The problem then becomes how do we choose $\mathbf{L}_i$ to be consistent with the known matrix of second derivatives $\mathbf{H}_i$. In other words, we have to show that at $\boldsymbol{\beta} = \boldsymbol{\beta}_i$

$$d_i + \frac{1}{2}\frac{\boldsymbol{\beta}^{\mathrm{T}}\mathbf{L}_i\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^2} = d_i + \frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\beta}_i)^{\mathrm{T}}\mathbf{H}_i(\boldsymbol{\beta}-\boldsymbol{\beta}_i)$$

to second order. The solution to this problem occurs when we let $\mathbf{L}_i = \|\boldsymbol{\beta}_i\|^2\mathbf{H}_i$ as we will now show.

To prove this, consider the function

$$g_i(\boldsymbol{\beta}) \equiv \frac{1}{2}\frac{\boldsymbol{\beta}^{\mathrm{T}}\mathbf{L}_i\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^2}.$$

After noting that

$$\begin{aligned} \frac{\partial}{\partial\beta_i}\left[\frac{1}{\|\boldsymbol{\beta}\|^k}\right] &= \frac{\partial}{\partial\beta_i}\frac{1}{(\sum_j \beta_j^2)^{k/2}} \\ &= -\frac{k}{2}\frac{2\beta_i}{(\sum_j \beta_j^2)^{k/2+1}} \\ &= -\frac{k\beta_i}{\|\boldsymbol{\beta}\|^{k+2}} \end{aligned}$$

we have that

$$\frac{\partial g_i}{\partial \beta_k} = \frac{[\mathbf{L}_i\boldsymbol{\beta}]_k}{\|\boldsymbol{\beta}\|^2} - \frac{\boldsymbol{\beta}^{\mathrm{T}}\mathbf{L}_i\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^4}\beta_k$$

and

$$\frac{\partial^2 g_i}{\partial \beta_k \partial \beta_l} = \frac{[\mathbf{L}_i]_{kl}}{\|\boldsymbol{\beta}\|^2} - \frac{2[\mathbf{L}_i\boldsymbol{\beta}]_k\beta_l}{\|\boldsymbol{\beta}\|^4} - \frac{2[\mathbf{L}_i\boldsymbol{\beta}]_l\beta_k}{\|\boldsymbol{\beta}\|^4} - \frac{\boldsymbol{\beta}^{\mathrm{T}}\mathbf{L}_i\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^4}\delta_{kl} + 4\frac{\boldsymbol{\beta}^{\mathrm{T}}\mathbf{L}_i\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^6}\beta_k\beta_l,$$

where δ_{kl} is the delta function. Thus if $\mathbf{H}_{g_i}(\boldsymbol{\beta}_i)$ is the Hessian of $g_i(\boldsymbol{\beta})$ at the location $\boldsymbol{\beta} = \boldsymbol{\beta}_i$, we have that

$$\mathbf{H}_{g_i}(\boldsymbol{\beta}_i) = \frac{\mathbf{L}_i}{\|\boldsymbol{\beta}_i\|^2} - \frac{2}{\|\boldsymbol{\beta}_i\|^4}[\mathbf{L}_i\boldsymbol{\beta}_i\boldsymbol{\beta}_i{}^{\mathrm{T}} + \boldsymbol{\beta}_i\boldsymbol{\beta}_i{}^{\mathrm{T}}\mathbf{L}_i] - \frac{\boldsymbol{\beta}_i{}^{\mathrm{T}}\mathbf{L}_i\boldsymbol{\beta}_i}{\|\boldsymbol{\beta}_i\|^4}\mathbf{I} + 4\frac{\boldsymbol{\beta}_i{}^{\mathrm{T}}\mathbf{L}_i\boldsymbol{\beta}_i}{\|\boldsymbol{\beta}_i\|^6}\boldsymbol{\beta}_i\boldsymbol{\beta}_i{}^{\mathrm{T}}.$$

If we choose that $\mathbf{L}_i = c\mathbf{H}_i$, where c is a scalar constant, then $\mathbf{L}_i\boldsymbol{\beta}_i = c\mathbf{H}_i\boldsymbol{\beta}_i = 0$ from before. Therefore, the Hessian $\mathbf{H}_{g_i}(\boldsymbol{\beta}_i)$ reduces to

$$\mathbf{H}_{g_i}(\boldsymbol{\beta}_i) = \frac{c\mathbf{L}_i}{\|\boldsymbol{\beta}_i\|^2}$$

at $\boldsymbol{\beta} = \boldsymbol{\beta}_i$. So if we set $c = \|\boldsymbol{\beta}_i\|^2$ then the approximation

$$d_i(\boldsymbol{\beta}) \sim d_i + \frac{1}{2}\frac{\boldsymbol{\beta}^{\mathrm{T}}\mathbf{L}_i\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^2} = d_i + \frac{1}{2}\frac{\boldsymbol{\beta}^{\mathrm{T}}(\|\boldsymbol{\beta}_i\|^2\mathbf{H}_i)\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^2}$$

has $\mathbf{H}_i$ as its Hessian at $\boldsymbol{\beta} = \boldsymbol{\beta}_i$ and is a generally more accurate approximation than

$$d_i(\boldsymbol{\beta}) \sim d_i + \frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\beta}_i)^{\mathrm{T}}\mathbf{H}_i(\boldsymbol{\beta} - \boldsymbol{\beta}_i).$$

Deciding Whether Two Segments are Part of the Same Trace

We now suppose that we have independently identified the best fits $\mathcal{C}_1 \equiv \mathcal{C}_{\boldsymbol{\beta}_1}$ and $\mathcal{C}_2 \equiv \mathcal{C}_{\boldsymbol{\beta}_2}$ to the traces $\mathcal{S}_1$ and $\mathcal{S}_2$ as per Chapter 4, and we next wish to decide whether they can be adequately fitted by the one curve.

First, however, we must move to a common coordinate setting for two segments, since both were fitted in their own coordinate setting with this setting optimized for numerical stability. Having specified the common coordinate system $\mathbf{x}''$ as in section 5.1.6, suppose that the transformations necessary to map from the individual coordinate settings are

$$\mathbf{x}'' = \alpha_i(\mathbf{x}'_i - \mathbf{u}_i), \qquad i = 1, 2.$$

These in turn induce a transformation to a common parameter setting for the curves, given by

$$\boldsymbol{\gamma}_i = \mathbf{A}_i\boldsymbol{\beta}_i, \qquad i = 1, 2$$

where the $\mathbf{A}_i$ are determined by (4.6). Therefore, for the model function defined by (4.13) and $\boldsymbol{\beta}_i$ by (5.3), the transformation matrix is given by

$$\mathbf{A}_i = \begin{bmatrix} 1 & v_i & v_i^2 & v_i^3 & v_i^4 & u_i \\ 0 & \frac{1}{\alpha_i} & \frac{2v_i}{\alpha_i} & \frac{3v_i^2}{\alpha_i} & \frac{4v_i^3}{\alpha_i} & 0 \\ 0 & 0 & \frac{1}{\alpha_i^2} & \frac{3v_i}{\alpha_i^2} & \frac{6v_i^2}{\alpha_i^2} & 0 \\ 0 & 0 & 0 & \frac{1}{\alpha_i^3} & \frac{4v_i}{\alpha_i^3} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\alpha_i^4} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{\alpha_i} \end{bmatrix} \tag{5.7}$$

where $\mathbf{u}_i = [u_i, v_i]^{\mathrm{T}}$.

Note that if merging of the traces $\mathcal{S}_1$ and $\mathcal{S}_2$ were possible, then we would have one parameter $\boldsymbol{\gamma}$ for a curve that adequately fits both traces. In this new coordinate setting the distance metric to be minimized for the joint fit is

$$\begin{aligned} d(\boldsymbol{\gamma}) &= \sum_{i=1}^{2}\sum_{n=1}^{N_i} \min_{\mathbf{x}'' \in \mathcal{C}_{\boldsymbol{\gamma}}} \|\mathbf{x}''_{i,n} - \mathbf{x}''\|^2 \\ &= \sum_{i=1}^{2} \alpha_i^2 d_i(\mathbf{A}_i^{-1}\boldsymbol{\gamma}) \end{aligned} \tag{5.8}$$

where $d_i(\boldsymbol{\beta})$ are the error metrics for the best fits to the individual traces, and $\mathbf{x}''_{i,n}$ is the n-th point of trace $\mathcal{S}_i$ in the common coordinate system.

At this point there are two options for proceding further, a direct approach and an indirect approach, to minimizing (5.8).

Direct Approach

In this approach we simply substitute in the previously determined second order approximations to $d_i(\boldsymbol{\beta})$, giving

$$d(\boldsymbol{\gamma}) \sim \sum_{i=1}^{2} \alpha_i^2 d_i + \frac{1}{2} \sum_{i=1}^{2} \alpha_i^2 \|\boldsymbol{\beta}_i\|^2 \frac{\boldsymbol{\gamma}^{\mathrm{T}} \mathbf{A}_i^{-1^{\mathrm{T}}} \mathbf{H}_i \mathbf{A}_i^{-1} \boldsymbol{\gamma}}{\|\mathbf{A}_i^{-1} \boldsymbol{\gamma}\|^2}. \tag{5.9}$$

We then choose $\boldsymbol{\gamma}$ to minimize this function.

Unfortunately, this approach has two disadvantages. Firstly, minimization of (5.9) is a nonstandard eigenvalue problem: it can be solved, but would require tweaking of the usual packages. Secondly, the $d(\boldsymbol{\gamma})$ of (5.9) is invariant to scaling but is not in the more convenient form of a function $d(\boldsymbol{\gamma}/\|\boldsymbol{\gamma}\|)$.

Indirect Approach

Returning to (5.8), we note that if we define

$$g_i(\boldsymbol{\gamma}) \equiv \alpha_i^2 d_i(\mathbf{A}_i^{-1} \boldsymbol{\gamma})$$

then at any point $\boldsymbol{\gamma} = \mathbf{A}_i \boldsymbol{\beta}$ we have that

$$\mathbf{H}_{g_i}(\boldsymbol{\gamma}) = \alpha_i^2 \mathbf{A}_i^{-1^{\mathrm{T}}} \mathbf{H}_i(\boldsymbol{\beta}) \mathbf{A}_i^{-1}.$$

(This follows from straightforward results on the effect of linear transformations on derivatives in multivariate calculus). Therefore in particular at the point $\boldsymbol{\gamma}_i = \mathbf{A}_i \boldsymbol{\beta}_i$ where we know the Hessian of $d_i(\boldsymbol{\beta})$ is $\mathbf{H}_i$ we have that the Hessian of g_i is

$$\mathbf{H}_{g_i}(\boldsymbol{\gamma}_i) = \alpha_i^2 \mathbf{A}_i^{-1^{\mathrm{T}}} \mathbf{H}_i \mathbf{A}_i^{-1}$$

and this satisfies

$$\mathbf{H}_{g_i}(\boldsymbol{\gamma}_i) \boldsymbol{\gamma}_i = \alpha_i^2 \mathbf{A}_i^{-1^{\mathrm{T}}} \mathbf{H}_i \mathbf{A}_i^{-1} \mathbf{A}_i \boldsymbol{\beta}_i = \alpha_i^2 \mathbf{A}_i^{-1^{\mathrm{T}}} \mathbf{H}_i \boldsymbol{\beta}_i = 0.$$

Therefore we could approximate $g_i(\boldsymbol{\gamma})$ by

$$g_i(\boldsymbol{\gamma}) \equiv g_i(\frac{\boldsymbol{\gamma}}{\|\boldsymbol{\gamma}\|}) \sim \alpha_i^2 d_i + \frac{1}{2} \alpha_i^2 \frac{\boldsymbol{\gamma}^{\mathrm{T}} \mathbf{A}_i^{-1^{\mathrm{T}}} \mathbf{H}_i \mathbf{A}_i^{-1} \boldsymbol{\gamma} \|\mathbf{A}_i \boldsymbol{\beta}_i\|^2}{\|\boldsymbol{\gamma}\|^2}$$

without any real loss of information: the Hessian of this function agrees with that of $d_i(\boldsymbol{\beta})$ at the minimum $\boldsymbol{\beta}_i$ and this is really all the information we have.

Following this approach would give an alternative approximation to $d(\boldsymbol{\gamma})$ of

$$d(\boldsymbol{\gamma}) \sim \sum_{i=1}^{2} \alpha_i^2 d_i + \frac{1}{2} \frac{\boldsymbol{\gamma}^{\mathrm{T}} (\sum_{i=1}^{2} \alpha_i^2 \|\mathbf{A}_i \boldsymbol{\beta}_i\|^2 \mathbf{A}_i^{-1^{\mathrm{T}}} \mathbf{H}_i \mathbf{A}_i^{-1}) \boldsymbol{\gamma}}{\|\boldsymbol{\gamma}\|^2}.$$

As a function of $\boldsymbol{\gamma}$ this can be easily minimized by choosing $\boldsymbol{\gamma}$ to be the smallest eigenvalue of the matrix

$$\mathbf{H} \equiv \sum_{i=1}^{2} \alpha_i^2 \|\mathbf{A}_i \boldsymbol{\beta}_i\|^2 \mathbf{A}_i^{-1^{\mathrm{T}}} \mathbf{H}_i \mathbf{A}_i^{-1}.$$

Regardless of which approach is used, having chosen the $\boldsymbol{\gamma}$ that minimizes an approximation to $d(\boldsymbol{\gamma})$, it remains to decide whether the estimated error in fitting both curves simultaneously is significantly greater than the sum of the errors incurred in fitting each separately. We do this by placing a statistical interpretation on the errors as follows.

Statistical Interpretation of Errors

First, for convenience, for any given point $\mathbf{x}$ define the best approximation to $\mathbf{x}$ on $\mathcal{C}$ by

$$\mathbf{x}_{\mathcal{C}} \equiv \arg\min_{\mathbf{y}\in\mathcal{C}} \|\mathbf{x}-\mathbf{y}\|^2.$$

Now for each trace segment $\mathcal{S}_i$ consider the residuals

$$\epsilon_{i,n} = x_{i,n} - [\mathbf{x}_{\mathcal{C}}]_{i,n}, \qquad n = 1,\ldots,2N_i.$$

If we suppose these to be Gaussian random variables with zero mean and unknown variance, then our best estimate for the variance σ_i^2 is

$$\sigma_i^2 \sim \frac{\alpha_i^2 d_i}{2N_i - 5}. \tag{5.10}$$

Note here that we are looking at the residuals in the common coordinate setting for both traces, and the denominator must be reduced by the number of free parameters used in the minimization (as $\mathcal{C}_{\boldsymbol{\beta}}$ is invariant under scaling of $\boldsymbol{\beta}$, although there are nominally 6 free parameters in $\boldsymbol{\beta}$ in practice there are only 5).

Now if the curve $\mathcal{C}_{\boldsymbol{\gamma}^*}$ defined by the parameter choice $\boldsymbol{\gamma}^*$ that minimizes $d(\boldsymbol{\gamma})$ fits both segments reasonably closely, we would expect that at worst the distribution of its residuals would be similar to the worst of the distributions of residuals after the best fit to each trace independently. Thus if we define

$$\sigma^2 = \max_i \sigma_i^2 \tag{5.11}$$

we would expect that $\frac{d(\boldsymbol{\gamma}^*)}{\sigma^2}$ should be $\chi^2_{2N_1+2N_2-5}$ distributed, and we can use a standard χ^2 test to decide whether $d(\boldsymbol{\gamma}^*)$ is significantly larger than anticipated. Using the χ^2 integral, we compute the probability $P(\chi^2_{2N_1+2N_2-5} > \frac{d(\boldsymbol{\gamma}^*)}{\sigma^2})$, and if this probability is less than 0.1, then we say that the distribution of errors of the estimated joint fit is not consistent with the two traces segments belonging to the same underlying trace. A tougher test would be to define

$$\sigma^2 = \frac{1}{2N_1+2N_2-5}\sum_{i=1}^{2}\alpha_i^2 d_i$$

and to use this value as the variance parameter to normalize $d(\boldsymbol{\gamma}^*)$ before making a $\chi^2_{2N_1+2N_2-5}$ test.

Two questions remain to be answered before the second order test can be used. They are, firstly, the value of d_i for each of the individual trace fits, and secondly, the Hessian at these fits must be determined. We will now consider how to determine d_i, the error in the individual fits.

Fitting Error

The ODRPACK fitting routine returns to the user the value of the fitting metric at its termination (any other reasonable fitting routine would also return such a value). It would be reasonable to expect that this is the value of d_i that should be used in the merge test, but this is not so for the following subtle reasons.

It is important to realize that the pixel locations in the ionograms are in error $\pm 1/2$ pixels in both x and y directions because of their quantized nature. Therefore, the error in the fit cannot have a variance less than the sum of variances of error in the pixels because of the uncertainty in their locations, even though the fitting routine may sometimes say otherwise. If the width of a pixel is p_i for trace i in its normalized local coordinate system, and assuming that the true pixel location is uniformly distributed across the entire square, then the variance of the error in each direction is

$$\sigma_i^2 = E[(x-\mu)^2] = \frac{p_i^2}{12}.$$

From section 4.4.1, the width of a pixel is given by $p_i = 1/(y_{\max_i} - y_{\min_i})$. So if trace i is N_i pixels long, then the error in the fit will be never less than $N_i p_i^2/6$. Therefore, if the fitting routine returns that the fitting error is d_{fit_i}, then the value of d_i we use in our χ^2 test for merge is given by

$$d_i = \max\{d_{\text{fit}_i}, N_i\frac{p_i^2}{6}\}. \tag{5.12}$$

The need for this was confirmed in experiments. Short traces could be fitted extremely closely by the fitting routine, but it was found that the error in this case for the individual fits bore little relationship with the error of a joint fit, even when it clearly should take place.

Estimating the Hessian

The ODRPACK routine purports to return the Hessian at the solution [23] to the user when desired. However, the matrix it returns is wrong in the implicit case because it is computed without regard to the constraints. For this reason it is necessary to compute an estimate of the Hessian oneself, using the method I present below.

From (4.9), the fitting problem for a single trace can be simply expressed as solving the minimization problem

$$\begin{aligned} &\min_{\beta_j, j=1,\ldots,L} \sum_{n=1}^{N} w_n(\delta_{x_n}^2 + \delta_{y_n}^2) \\ &\text{subject to} \qquad f(x_n + \delta_{x_n}, y_n + \delta_{y_n}, \boldsymbol{\beta}) = 0, \qquad n = 1, \ldots, N \end{aligned} \tag{5.13}$$

where L is the number of free model parameters and N is the number of pixels. Let

$$\mathbf{z} = [\delta_{x_1}, \delta_{y_1}, \delta_{x_2}, \delta_{y_2}, \ldots, \delta_{x_N}, \delta_{y_N}]^{\mathrm{T}}$$

and then (5.13) is a special case of the problem

$$\begin{aligned} &\min_{\beta_j, j=1,\ldots,L} \sum_{k=1}^{2N} t_k^2 z_k^2 \\ &\text{subject to} \qquad f_n(\mathbf{z}, \boldsymbol{\beta}) = 0, \quad n = 1, \ldots, N \end{aligned}$$

where $t_{2i-1} = t_{2i} = \sqrt{w_i}$, $f_n(\mathbf{z}, \boldsymbol{\beta})$ is a collection of functions formed from $f(x_n + \delta_{x_n}, y_n + \delta_{y_n}, \boldsymbol{\beta})$ by substituting in each coordinate pair (x_n, y_n) and re-indexing onto the relevant elements of $\mathbf{z}$ in each case. Now let us assume that the minimum of this problem occurs at $\mathbf{z}^*$, $\boldsymbol{\beta}^*$. Then,

$$\begin{aligned} f_n(\mathbf{z}, \boldsymbol{\beta}) &\sim f_n(\mathbf{z}^*, \boldsymbol{\beta}^*) + \sum_k \frac{\partial f_n}{\partial z_k}(\mathbf{z}^*, \boldsymbol{\beta}^*)(z_k - z_k^*) + \sum_j \frac{\partial f_n}{\partial \beta_j}(\mathbf{z}, \boldsymbol{\beta}^*)(\beta_j - \beta_j^*) \\ &\sim 0 + J_{z_k}(\mathbf{z} - \mathbf{z}^*) + J_{\beta_j}(\boldsymbol{\beta} - \boldsymbol{\beta}^*). \end{aligned}$$

For all n together, we get the matrix equation

$$\mathbf{f} = \mathbf{0} + \mathbf{J}_{\mathbf{z}}(\mathbf{z} - \mathbf{z}^*) + \mathbf{J}_{\boldsymbol{\beta}}(\boldsymbol{\beta} - \boldsymbol{\beta}^*).$$

If $\boldsymbol{\beta}$ is set close to $\boldsymbol{\beta}^*$, and $\mathbf{z}$ is adjusted to restore the constraint, then

$$\mathbf{J}_{\mathbf{z}}\mathbf{z} = \mathbf{J}_{\mathbf{z}}\mathbf{z}^* - \mathbf{J}_{\boldsymbol{\beta}}(\boldsymbol{\beta} - \boldsymbol{\beta}^*).$$

The required minimum is the solution $\mathbf{z}$ that minimizes $\sum t_k^2 z_k^2$. To eliminate the weights, let $\mathbf{D} = \mathrm{diag}(t_i)$, $\mathbf{r} = \mathbf{D}\mathbf{z}$, and $\mathbf{r}^* = \mathbf{D}\mathbf{z}^*$. Then,

$$\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1}\mathbf{r} = \mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1}\mathbf{r}^* - \mathbf{J}_{\boldsymbol{\beta}}(\boldsymbol{\beta} - \boldsymbol{\beta}^*).$$

The solution that minimizes $\|\mathbf{r}\|^2$ is

$$\mathbf{r} = (\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1})^{+}\{\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1}\mathbf{r}^* - \mathbf{J}_{\boldsymbol{\beta}}(\boldsymbol{\beta} - \boldsymbol{\beta}^*)\}$$

and therefore

$$\begin{aligned} \|\mathbf{r}\|^2 = \; & \mathbf{r}^{*\mathrm{T}}\mathbf{D}^{-1}\mathbf{J}_{\mathbf{z}}^{\mathrm{T}}(\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1})^{+\mathrm{T}}(\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1})^{+}\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1}\mathbf{r}^* \\ & -2\mathbf{r}^{*\mathrm{T}}\mathbf{D}^{-1}\mathbf{J}_{\mathbf{z}}^{\mathrm{T}}(\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1})^{+\mathrm{T}}(\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1})^{+}\mathbf{J}_{\boldsymbol{\beta}}(\boldsymbol{\beta} - \boldsymbol{\beta}^*) \\ & +(\boldsymbol{\beta}^{\mathrm{T}} - \boldsymbol{\beta}^{*\mathrm{T}})\mathbf{J}_{\boldsymbol{\beta}}^{\mathrm{T}}(\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1})^{+\mathrm{T}}(\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1})^{+}\mathbf{J}_{\boldsymbol{\beta}}(\boldsymbol{\beta} - \boldsymbol{\beta}^*). \end{aligned}$$

Now the second term is equal to zero because when $\boldsymbol{\beta} - \boldsymbol{\beta}^* = 0$, $\|\mathbf{r}\|^2$ is minimized. From the third term, we obtain that the Hessian of the distance metric at the solution is

$$\mathbf{H} = 2\mathbf{J}_{\boldsymbol{\beta}}^{\mathrm{T}}(\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1})^{+\mathrm{T}}(\mathbf{J}_{\mathbf{z}}\mathbf{D}^{-1})^{+}\mathbf{J}_{\boldsymbol{\beta}}.$$

In the special case,

$$\mathbf{J_z D}^{-1} = \begin{bmatrix} \frac{\partial f}{\partial x}(x_1+\delta^*_{x_1}, y_1+\delta^*_{y_1}, \boldsymbol{\beta}^*)/t_1 & \frac{\partial f}{\partial y}(x_1+\delta^*_{x_1}, y_1+\delta^*_{y_1}, \boldsymbol{\beta}^*)/t_1 & 0 \\ 0 & 0 & \frac{\partial f}{\partial x}(x_2+\delta^*_{x_2}, y_2+\delta^*_{y_2}, \boldsymbol{\beta}^*)/t_2 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & \cdots & 0 \\ \frac{\partial f}{\partial y}(x_2+\delta^*_{x_2}, y_2+\delta^*_{y_2}, \boldsymbol{\beta}^*)/t_2 & \cdots & 0 \\ \vdots & \cdots & \vdots \\ 0 & \cdots & \frac{\partial f}{\partial y}(x_N+\delta^*_{x_N}, y_N+\delta^*_{y_N}, \boldsymbol{\beta}^*)/t_N \end{bmatrix}$$

and

$$[\mathbf{J}_{\boldsymbol{\beta}}]_{ij} = \frac{\partial f}{\partial \beta_j}(x_i+\delta^*_{x_i}, y_i+\delta^*_{y_i}, \boldsymbol{\beta}^*).$$

Note that for any matrix $\mathbf{J}$ with the structure

$$\mathbf{J} = \begin{bmatrix} a_1 & b_1 & & & & \\ & & a_2 & b_2 & & \\ & & & & \ddots & \\ & & & & a_N & b_N \end{bmatrix}$$

it is easily confirmed that the Moore-Penrose inverse of this matrix is

$$\mathbf{J}^+ = \begin{bmatrix} a_1/(a_1^2+b_1^2) & & & & & \\ b_1/(a_1^2+b_1^2) & & & & & \\ & a_2/(a_2^2+b_2^2) & & & & \\ & b_2/(a_2^2+b_2^2) & & & & \\ & & & & \ddots & \\ & & & & a_N/(a_N^2+b_N^2) & \\ & & & & b_N/(a_N^2+b_N^2) & \end{bmatrix}$$

and we obtain that

$$\mathbf{J}^{+\mathrm{T}}\mathbf{J}^+ = \begin{bmatrix} 1/(a_1^2+b_1^2) & & & \\ & 1/(a_2^2+b_2^2) & & \\ & & \ddots & \\ & & & 1/(a_N^2+b_N^2) \end{bmatrix}.$$

Using this result, we obtain that the Hessian $\mathbf{H}$ is given by

$$[\mathbf{H}]_{ij} = 2\sum_{n=1}^{N} \left[\frac{w_i \frac{\partial f}{\partial \beta_i}\frac{\partial f}{\partial \beta_j}}{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \Bigg|_{(x_n+\delta^*_{x_n}, y_n+\delta^*_{y_n}, \boldsymbol{\beta}^*)} \right]. \tag{5.14}$$

5.1.8 Test 7: Extrapolation

The extrapolation test is the test of last resort because it is so computationally expensive. This test computes how well the extension of one trace fits the pixels of another (figure 5.8). It does this by computing the closest point of the fit of the longer trace to the pixels of the shorter using an optimization routine. The distance computed is compared with the sum of scaled versions of the error from the individual fits using a χ^2 test as is done for the second order test.

5.1.9 Other Tests

Many other possible tests could be designed for merging. I will discuss two promising and powerful tests here that have not currently been implemented, but are based on observations of the behaviour of traces in the merging routine.

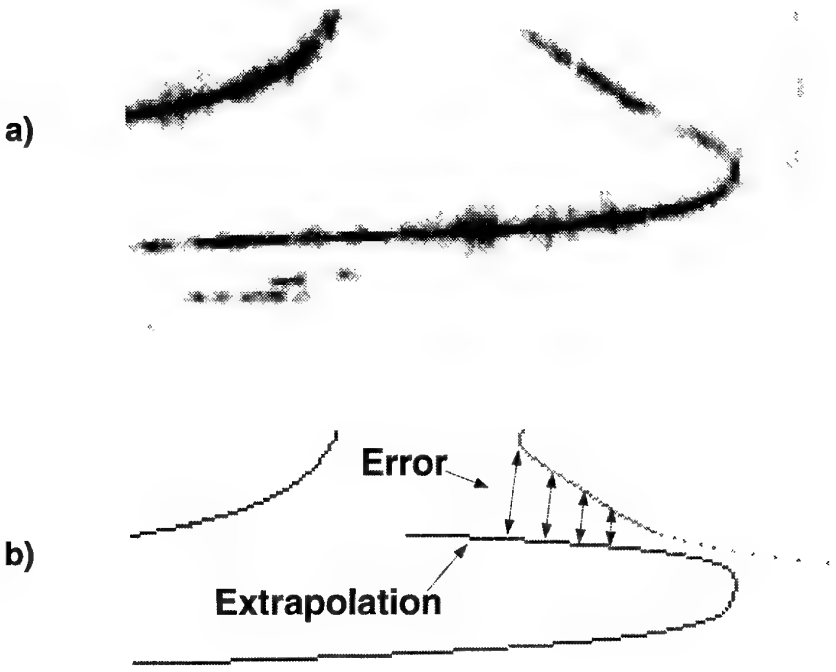


Figure 5.8: Test 7, the extrapolation test, examines how well the fit to the longer trace extrapolates to fit the shorter one.

Hough Test

One promising idea is to use a Hough transform [18] for detecting lines, and is based upon the following observation. Traces in ionograms are smoothly changing curves, which look on a small scale like straight lines away from the junction frequency. We can use this behaviour to design a test that can cope with breaks in the traces that occur along straight sections using the Hough transform (figure 5.9). By taking the Hough transform of the end sections of the trace fragments (of the filtered ionograms) we can determine the best possible straight line fit the ends. If these straight lines fit well the thinned skeletons of the other trace then the traces should be merged.

This test could be extended so that not only does it indicate when a merge is possible because the two straight line fits are almost identical, but it could also be used to eliminate those merge candidates where the line fits do not intersect one another in an acceptable way. By acceptable I mean that the point of intersection must be close enough to the region of the two traces (figure 5.10).

This Hough test will eliminate all the possible merges that the horizontals test already eliminates, but it would be practical to keep both because the horizontals test is very fast and by running it first some computational saving could result.

Intersection Test

There is one property of traces that has not yet been exploited in any of the merge tests that have been described so far. That is, no trace of a single propagation mode ever crosses over a trace of a different mode. This principle could be used to construct a test that would eliminate many more possible merges in the following way. Perhaps the simplest way to implement this would be to join a line between the closest end points of the two candidate traces and if this line intersects any other trace then the two candidates should not be joined (figure 5.11).

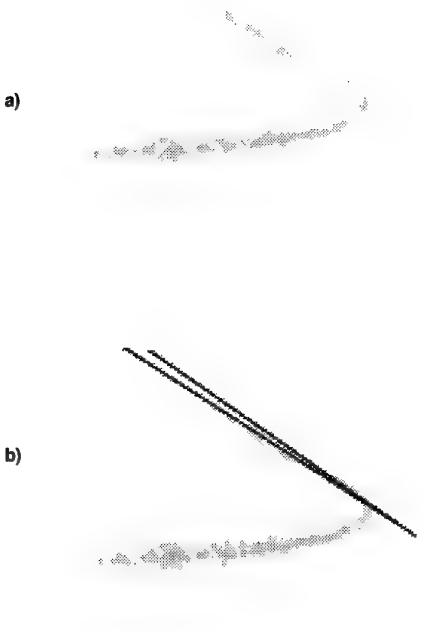


Figure 5.9: A merge test based on the Hough transform. The two trace fragments in (a) have been fitted at their ends with straight lines using the Hough transform, and these lines have been plotted on the filtered traces in (b).

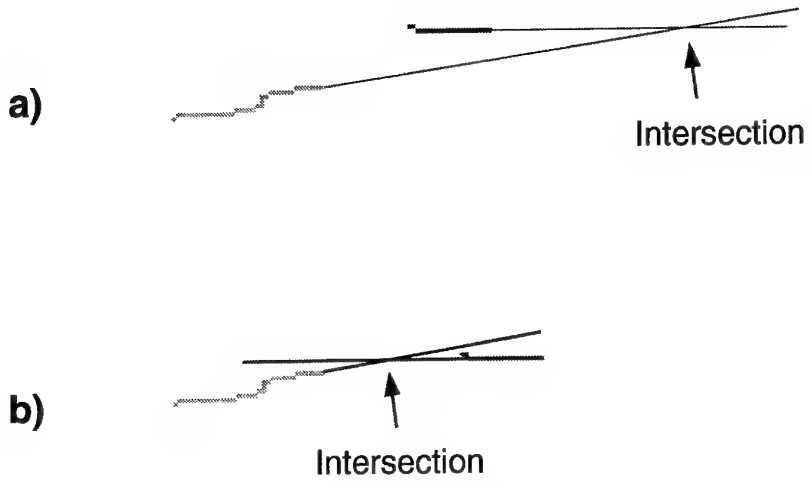


Figure 5.10: An extension of the Hough test for merging. In (a) the intersection point between the straight line fits to the ends of the two traces is too far away from the region of the two trace for a merge to be possible. The closeness of the intersection point in (b) indicates that a merge is possible in this case.

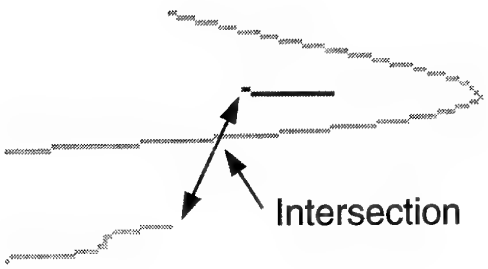


Figure 5.11: If two traces have another trace that passes between them then they should not be merged. This principle is the basis of the proposed intersection test.

5.2 Combining the Results

Currently the battery of merge tests just described is implemented as an ordered sweep through all the traces. Firstly, trace 1 is considered against trace 2 as candidates for merging. Then each of the tests are performed on these two traces in the order presented until one undicates conclusively either that the two traces should or should not be merged or no more tests remain to be performed in which case the traces are left as separate. If the tests indicate that 1 and 2 should be merged, then these two traces are eliminated from further consideration and traces 2 and 4 are tested for merging. If not, then trace 1 is tested against trace 3 using all the merged tests in order, and so on.

There are a number of points worth noting here. Firstly, the ordered treatment of traces above assumes that a trace will only be broken into at most two pieces. Rarely this will not be true however, and it may be necessary to consider applying these tests to more than two traces. The tests can be modified to achieve this, especially the second order test. Secondly, only tests 6 and 7 actually indicate that a merge should occur — all the others indicate either that it may be possible or that it is *not* possible. These two tests also give a good initial guess for fitting the two traces jointly. The fitting routine is called with this initial guess, as per the previous chapter, and the feature vectors from the ionogram fits are changed to reflect the fact the two trace fragments are now represented by the one fit. Lastly, it would be better to treat the collection of tests as a fusion problem rather than as an arbitrary preferential ordering. This approach may offer some advantages over the current approach, but it has not been investigated because of time limitations.

5.3 Results

The results of the merging procedure upon the selected branches of the fitted hand-trimmed ionograms shown in figures 5.14–5.17 can be seen in figures 5.18–5.21. (The traces that were pointed out so that the merged traces are not obscured.) The places where traces have been merged is indicated by arrows. Qualitatively, these merging results are good — no traces have been erroneously merged, nor are there any obvious cases where traces should have been merged which weren't.

However, there are problems with the estimation of the variance σ^2. To get these good results I found it necessary to add additional scaling to the estimate of the variance in the residuals for longer traces. For traces less than 60 pixels in length, no modification was required. But for traces over 160 pixels, σ^2 was enlarged by a factor of 10, and lengths between 60 and 160 were scaled to obtain a linear characteristic in this region. This seems to be the result of correlation in the residuals, as the following shows.

Figure 5.12 depicts the residuals along the x axis of a fit to a trace of 171 pixels. It is clear from this figure that the residuals hardly satisfy the assumption that they are normally distributed and uncorrelated. The presence of linear ramps indicates some sort of correlation structure.

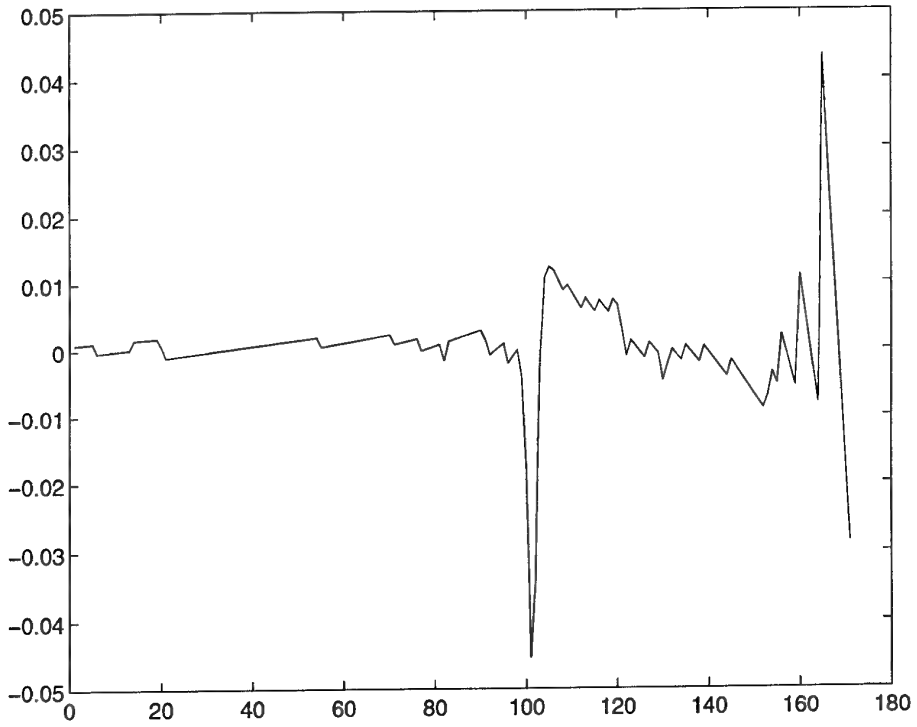


Figure 5.12: A plot of the residuals along the x-axis of a fit to a trace of 171 pixels. Note the piece-wise linear pattern of these errors, which are due to the quantized nature of the pixel locations from the fit which is locally linear.

Let the residuals in x be denoted by $\epsilon_{x,n}$, and similarly for y. In estimating the variance in section 5.1.7, we have assumed that the residuals $\epsilon_{x,n}, \epsilon_{y,n}$ are uncorrelated, but this is not true. We assume here now only that there is no cross-correlation between $\epsilon_{x,n}$ and $\epsilon_{y,n}$, but that they can be correlated within themselves, and that the residuals are stationary and have zero mean. Therefore,

$$E[\epsilon_{x,n}] = 0$$

and

$$E[\epsilon_{x,n}\epsilon_{x,m}] = c_{|n-m|}$$

If we compute the autocorrelation sequence by

$$c_{mn} \equiv c_{|n-m|} \equiv c_r = \frac{1}{N-r} \sum_{i=0}^{N-|r|-1} \epsilon_{x,i}\epsilon_{x,i+r}$$

then the resulting covariance matrix $[\mathbf{C}]_{mn} = c_{mn}$ will indicate the degree to which the estimate of σ^2 in (5.10,5.11,5.12) will deviate from the true value. The unbiased estimate of the autocorrelation sequence of the residuals in figure 5.12 is shown in figure 5.13. From this we can see that the residuals are positively correlated out to 4 pixels and at a range of 5–8 pixels they are negatively correlated. This indicates that there is indeed a problem with the estimate of σ^2 in (5.10,5.11,5.12). In simple terms, the sequence of residuals will thus act as if we have only one in four of the samples with a resulting smaller sample size, and the estimate of σ^2 should reflect this. These effects will be more pronounced in longer traces, as observed in the need for a fudge factor mentioned above.

5.4 Feature Extraction: the Output

The output of all the feature extraction processing that I have described in this and the previous chapters is a single feature vector for each trace, each of which corresponds to a single and distinct propagation

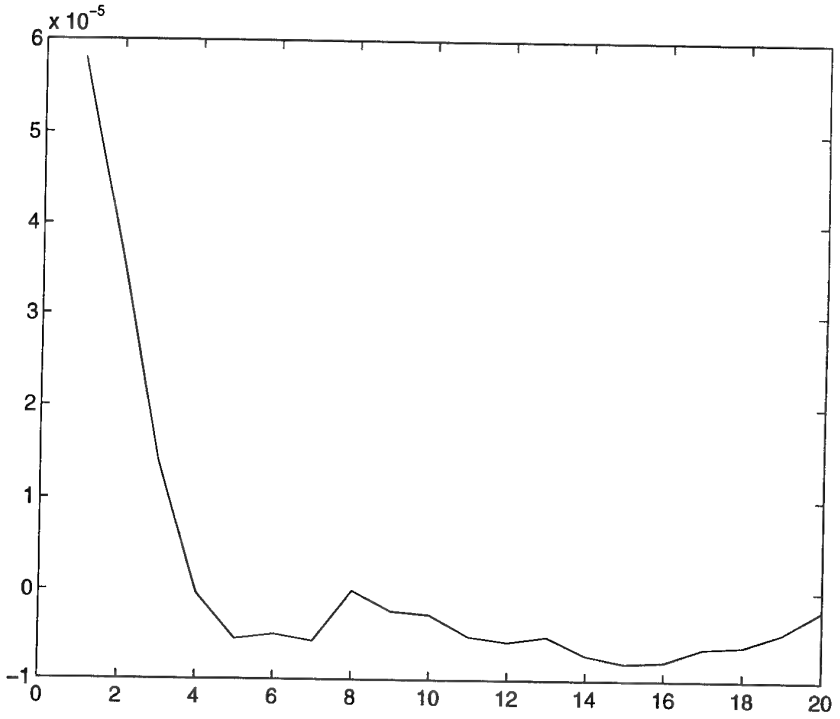


Figure 5.13: A plot of the autocorrelation sequence of residuals along the x-axis of a fit to a trace of 171 pixels. The sequence has been truncated at 20 because beyond this correlation length the estimate is not an accurate one due to the length and nonstationary effects.

mode. But what is the feature vector? The feature vector is taken from the parameter values $\boldsymbol{\beta}$ that describe the best fit of the model to each of the traces. To this is added information about location of the traces, and this is obtained from the x and y range of the trace.

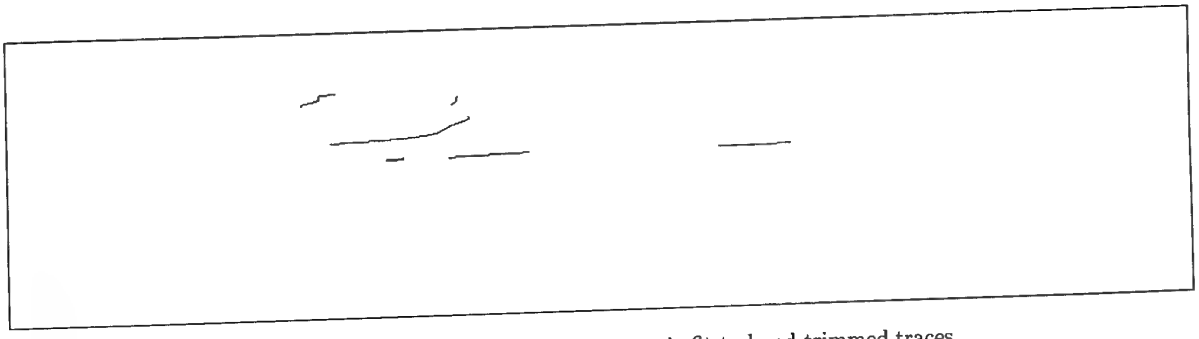


Figure 5.14: Ionogram 1 — implicit multivariate quartic fit to hand-trimmed traces.

Figure 5.15: Ionogram 2 — implicit multivariate quartic fit to hand-trimmed traces.

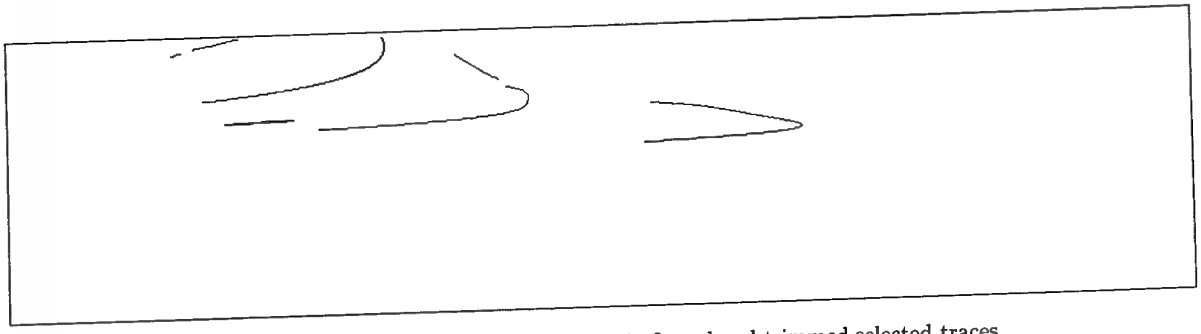


Figure 5.16: Ionogram 3 — implicit multivariate quartic fit to hand-trimmed selected traces.

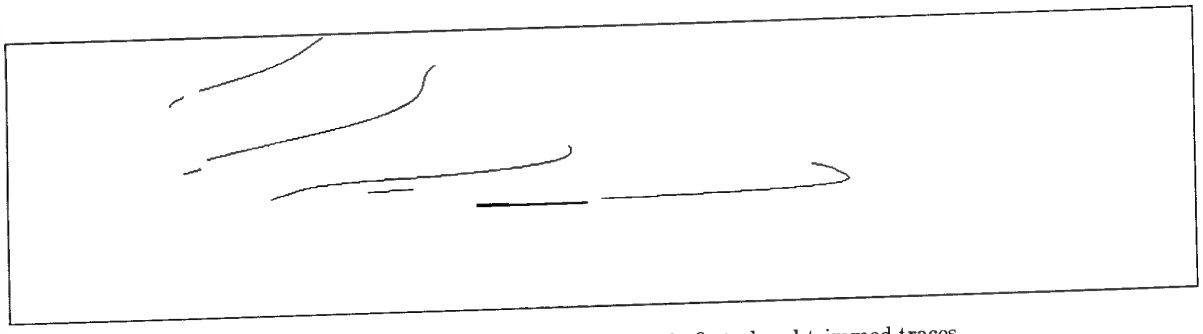


Figure 5.17: Ionogram 4 — implicit multivariate quartic fit to hand-trimmed traces.

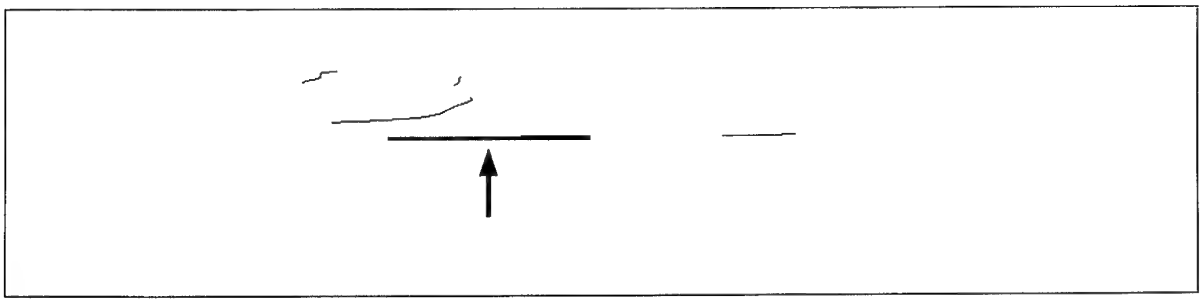


Figure 5.18: Ionogram 1 — Merge result of hand-trimmed traces.

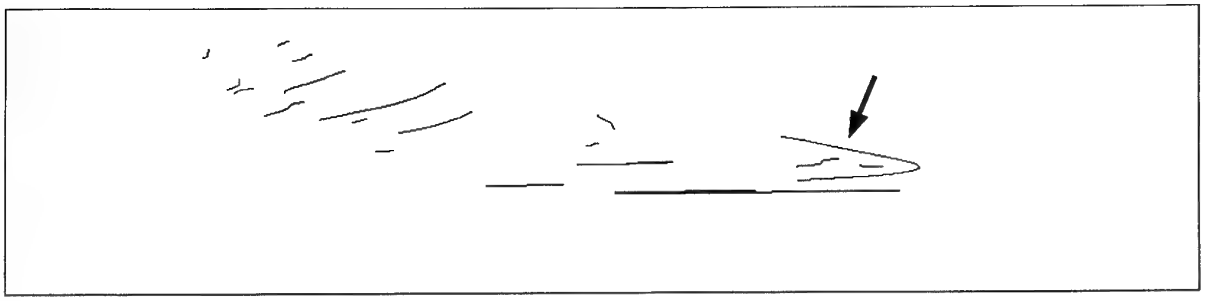


Figure 5.19: Ionogram 2 — Merge result of hand-trimmed traces.

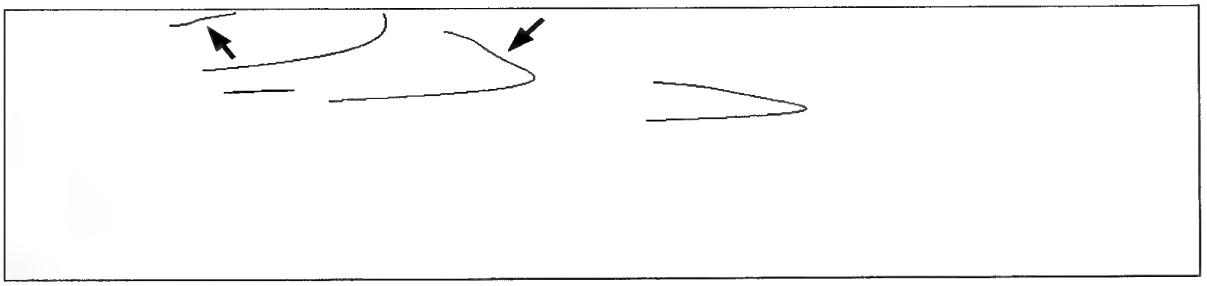


Figure 5.20: Ionogram 3 — Merge result of hand-trimmed selected traces.

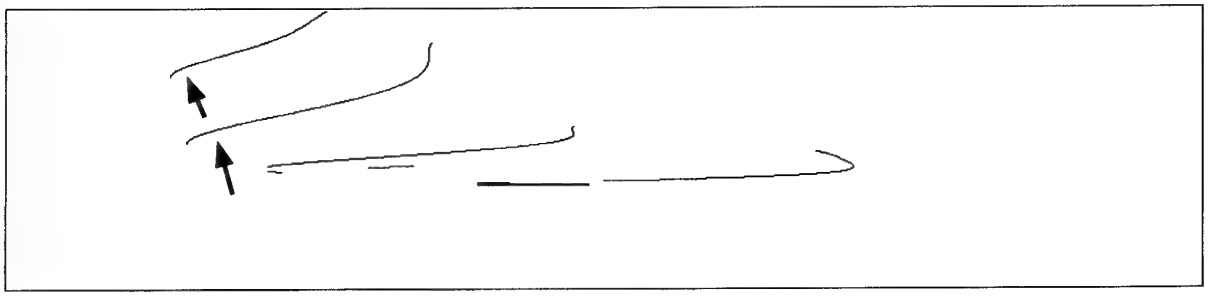


Figure 5.21: Ionogram 4 — Merge result of hand-trimmed traces.

Chapter 6

Mode Identification and Measurement

In this chapter the final stages of the autoscaling system are presented. These are, firstly, a tracking stage that is designed to assist mode identification in circumstances where one or more of the simple modes are missing from the ionogram because of various ionospheric effects. Secondly, a mode identification stage which makes use of an ionospheric model to identify the propagation modes of each trace. Lastly, possible further measurements are outlined. The flow of these stages in the autoscaling system is represented in figure 6.1.

6.1 Time-Series Tracking

The motivation for the time-series tracking stage is that ionograms are collected at regular intervals during which they usually do not change a great deal. Therefore, information gleaned from one ionogram can be used to make the task of identifying the modes of successive ionograms much easier. An example of a sequence of ionograms is shown in figure 6.2.

In figure 6.2, in the third ionogram of the sequence the 1-hop is almost completely missing, and what remains is unlikely to survive filtering. This absence could present difficulties to a mode identification scheme that did not realize when this has occurred. However, it is rather obvious when the modes are tracked across successive ionograms.

The time-series tracking could be achieved by a simple matching of the feature vectors in successive ionograms. In figure 6.3, it can be seen that there is an excellent correspondence between the shape of merged fits of successive ionograms that are due to the same propagation mode, even though they may move around in group-delay and frequency. Therefore, the feature vectors determined from the previous processing stages is ideal as an input to the tracking stage. This approach has beneficiary side effects demonstrated in figure 6.3(b): sometimes fragments of traces that it has not been possible to join up can be identified using the best possible match from the previous ionogram.

The first step in matching features across successive ionograms is to transform $\boldsymbol{\beta}$ from each of the fits into the same coordinate system using (5.7) so that the fits being compared are at the same scale. The best match can be made invariant under translation in the following way. Let $\boldsymbol{\beta}_i = [\beta_{i,0}, \ldots, \beta_{i,5}]^{\mathrm{T}}$ and $\boldsymbol{\beta}_j$ be the two model vectors from the fits to the traces of interest, and $\{x_{i_{\min}} \leqslant x_i \leqslant x_{i_{\max}}, y_{i_{\min}} \leqslant y_i \leqslant y_{i_{\max}}\}$, $\{x_{j_{\min}} \leqslant x_j \leqslant x_{j_{\max}}, y_{j_{\min}} \leqslant y_j \leqslant y_{j_{\max}}\}$ be their coordinates, respectively. Equation (5.7) is used to define the transformation matrix $\mathbf{A}$ and (5.1) with $\alpha = 1$ defines the coordinate transform $\mathbf{u} = (u, v)$. Then the best match can be found by solving

$$\begin{aligned} &\min_{u,v} \|\boldsymbol{\beta}_i - \mathbf{A}\boldsymbol{\beta}_j\|^2 \\ &\text{subject to} \qquad x_{i_{\min}} < x_{j_{\max}} \\ &\qquad\qquad\quad x_{j_{\min}} < x_{i_{\max}} \\ &\qquad\qquad\quad y_{i_{\min}} < y_{j_{\max}} \\ &\qquad\qquad\quad y_{j_{\min}} < y_{i_{\max}}. \end{aligned}$$

After finding the best match between each pair of trace fits, a search is performed to find the best

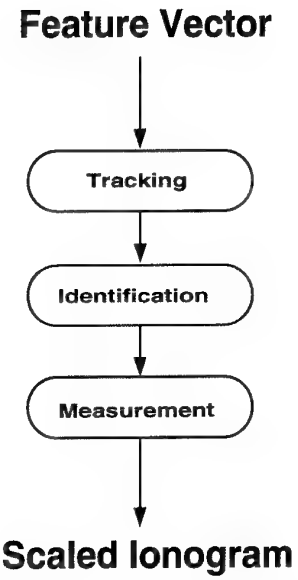


Figure 6.1: The last few stages of the autoscaling system takes a feature vector from the fitting stage as input and includes a tracking stage to cater for missing 1-hop traces, a mode identification stage, and then a final measurement stage.

possible set of these matches between the features subject to the condition that the translation of a particular propagation mode's trace from one ionogram to the next should be similar to that of other traces (*i.e.*, the black arrows from figure 6.3(a) to 6.3(b) should all point in roughly the same direction). As a possible extension to this approach, a Kalman filter could be used to predict the location of each trace of a particular propagation mode in the next ionogram to narrow down the number of possible comparisons between trace fits.

6.2 Mode Identification

The purpose of mode identification is to determine the hop number of traces that are due to simple propagation modes through the ionosphere. This is the second last stage in the overall autoscaling system, and the first one that will produce quantitative results for the user.

The mode identification stage is broken up into a number of smaller steps (figure 6.4). The first of these determines which trace in the fitted ionogram is the 1-hop trace. Mostly this will be the right-most trace except mainly in two special circumstances. Sometimes when the ionosphere contains a strong sporadic-E layer and the path length from transmitter to receiver is short a trace that bounces off this layer may extend in frequency out past the 1-hop trace. (This situation occurs in HFRD's FMS oblique ionograms [17].) However, it is possible to identify traces due to sporadic-E layers in group-delay using simple geometric considerations and thereby eliminate them. Secondly, the 1-hop trace may be missing from the ionogram, but the tracking stage of the previous section has been designed to recognize when this occurs. If the 1-hop trace is missing, then we continue the processing using the 2-hop trace instead and we must make some corresponding changes later on. Thus, after eliminating the two special cases, determining the 1-hop trace fit (or substituting the 2-hop) should be quite straight forward.

The next step assumes that we have a reasonable ionospheric model for oblique ionogram traces that can be used to predict the higher hops from knowledge of the 1-hop. Given then that we have a fit for the 1-hop trace (ignoring for a moment the complication of substituting a 2-hop for a non-existent 1-hop), we now have to find the parameters for this simplified ionospheric model so that there is a good correspondence between the 1-hop trace of the model and the 1-hop fit. This is not a trivial procedure because of the complexity of even a simplified ionospheric trace model as we will see.

Once a good fit of the ionospheric model's 1-hop (or 2-hop) to the appropriate trace of the ionogram has been obtained, the next step is to determine where this model predicts the higher order hops will

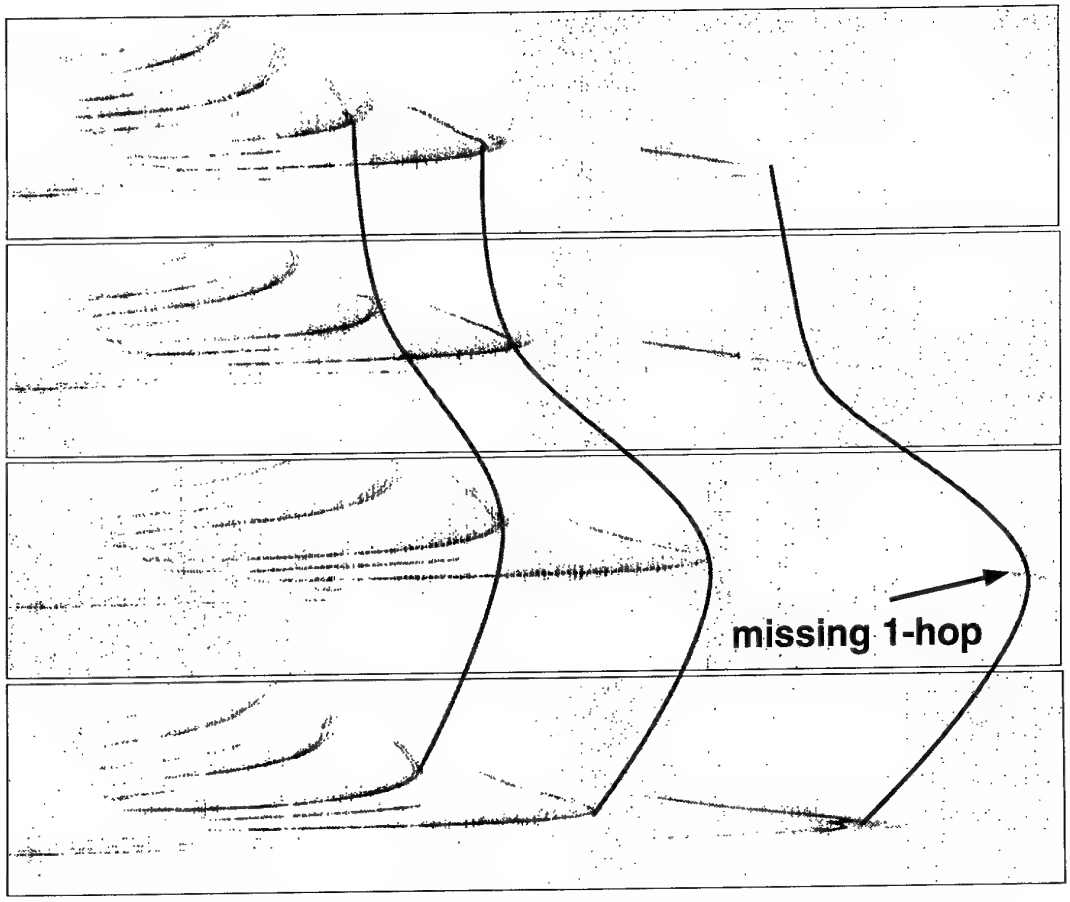


Figure 6.2: Shown here is a sequence of four successive ionograms, with the drift of the junctions for the first three simple modes noted by the wavy lines. Note that in the third ionogram the 1-hop is almost completely missing, and what remains is unlikely to survive filtering. This absent 1-hop trace could present difficulties to a mode identification scheme that did not realize when this has occurred, but it is rather obvious when the modes are tracked across successive ionograms.

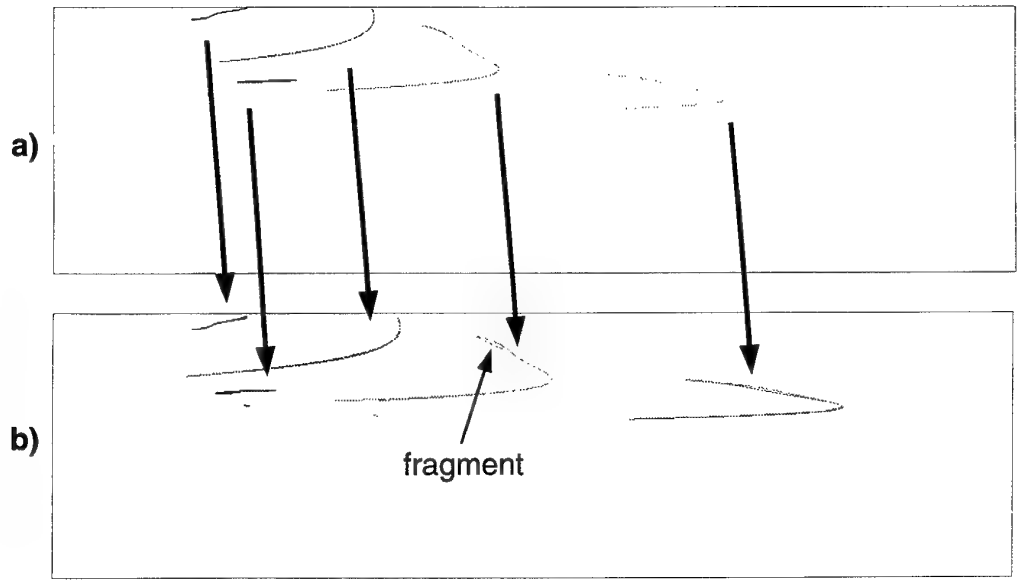


Figure 6.3: The merged fits of two successive ionograms are depicted in (a) and (b). In addition, in (b), the merged fits of (a) have been overlaid, using a simple translation, onto the corresponding fits of (b) showing that successive ionograms often do not change a great deal in character and that there is an excellent correspondence between the shape of the fits of the corresponding propagation mode. This characteristic can be used to assist in identifying the propagation modes of an ionogram given the hop numbers for the traces in the previous ionogram. In addition, a fragment in (b) that was not identified as belonging to a longer trace by the merging procedure has been identified by its close correspondence to the more extensive fit of the previous ionogram.

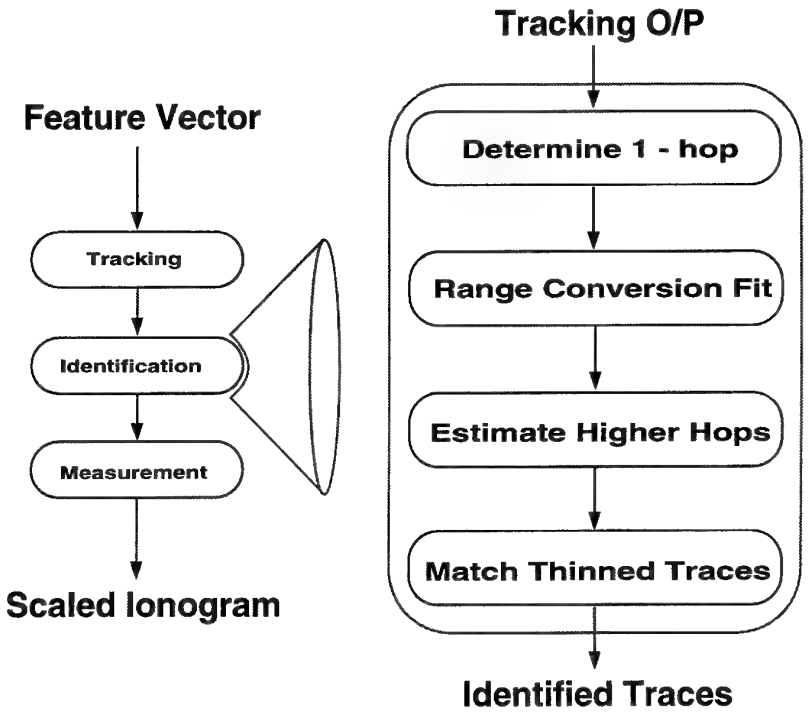


Figure 6.4: The intermediate stages in mode identification.

occur. Then, lastly, these trace predictions are cross-matched against the other fitted traces in the ionogram — the correspondences that occur will indicate the propagation mode of the important ionogram trace fits. Any traces that are not identified by this procedure will be due to complex propagation modes and are of lesser interest.

6.2.1 Ionospheric Model

Let us now consider the ionospheric model in more detail. The model is due to Dr Ken Lynn [31] and was produced by range converting a vertical incidence ionogram, based on a single parabolic layer, to an oblique incidence one. The model firstly defines the equation for a trace of a vertical ionogram derived from a parabolic electron density as

$$h_1 = h_0 + \frac{b}{2} f_1 \ln\left[\frac{1+f_1}{1-f_1}\right] \tag{6.1}$$

where h_0 is the height of base of ionosphere (200–300 km), h_1 is the vertical reflection height, b is the distance from base to peak electron density heights (10–150 km), and f_1 is a parameter of the model with range $0 < f_1 < 1$. Using conversion formulae for sky range [31], the equivalent oblique ionogram's time-delay t (in milliseconds) is given by

$$t = \frac{1}{150}\left[a^2 + (a+h_1)^2 - 2a(a+h_1)\cos\left(\frac{d}{2a}\right)\right]^{1/2} \tag{6.2}$$

and its frequency f by

$$f = \frac{f_s f_1}{\left[1 - \left(\frac{a}{150t}\sin\left(\frac{d}{2a}\right)\right)^2\right]^{1/2}} \tag{6.3}$$

where a is the radius of the earth (6371 km), d is the path length (distance of transmitter to receiver (300–6500 km)), and f_s a frequency scaling parameter. Higher hops can be calculated by dividing path length by the hop number and multiplying the resulting time-delay by the same factor.

Substituting (6.1) into (6.2) and (6.3) and including the hop number denoted by n, the resulting model for an oblique ionogram trace is given by the following parametric equations with parameter $0 < f_1 < 1$ and variables h_0, b and f_s:

$$\begin{aligned} t(f_1) &= \frac{n}{150}\left[a^2 + \left(a + h_0 + \frac{b}{2} f_1 \ln\left[\frac{1+f_1}{1-f_1}\right]\right)^2 - 2a\left(a + h_0 + \frac{b}{2} f_1 \ln\left[\frac{1+f_1}{1-f_1}\right]\right)\cos\left(\frac{d}{2na}\right)\right]^{1/2} \\ f(f_1) &= \frac{f_s f_1}{\left[1 - \frac{a^2 \sin\left(\frac{d}{2na}\right)^2}{a^2 + \left(a + h_0 + \frac{b}{2} f_1 \ln\left[\frac{1+f_1}{1-f_1}\right]\right)^2 - 2a\left(a + h_0 + \frac{b}{2} f_1 \ln\left[\frac{1+f_1}{1-f_1}\right]\right)\cos\left(\frac{d}{2na}\right)}\right]^{1/2}} \end{aligned} \tag{6.4}$$

An example of the traces that this model produces for a range of hop numbers is depicted in figure 6.5.

6.2.2 Range Conversion Fit

We now have a suitable ionospheric model with which to predict the location of higher hops given a good fit to the 1-hop (or 2-hop) just by varying n in (6.5). The problem that has to be solved is how to find a set of values for h_0, b, and f_s, given the feature vector for this one trace.

The first step in such a procedure is to compute an initial guess for these three parameters that can be further refined by an optimization procedure. We can construct a good initial guess from the following. Firstly, note that the minimum time-delay in the oblique trace will be set by h_0, and secondly the sharpness of the nose will be determined by b — the smaller the sharper. So then the first step in determining an initial guess is to set h_0 so that the model's y-value is equal to the minimum y value of the 1-hop trace. Next, we set b to be equal to a typical value, say 100 km, because we will allow the optimization procedure to adjust this value. We are interested here only in seeing the initial guess place the trace in the correct region. Thirdly, we set the scaling factor f_s to be a value such that the junction frequency of the ionospheric trace model (the right-most frequency at the nose) corresponds to the right

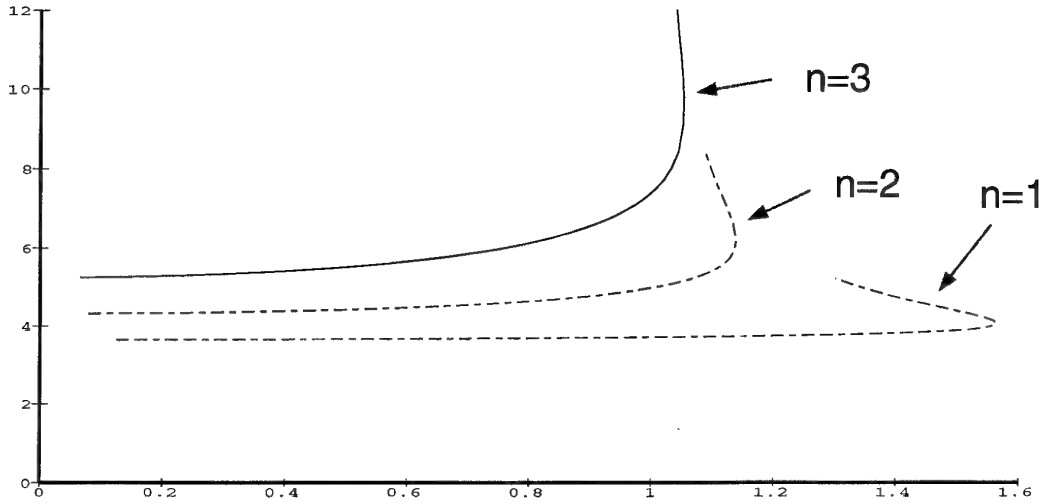


Figure 6.5: A example plot of 1-hop, 2-hop and 3-hop traces produced by sweeping $0 < f_1 < 1$ and setting $n = 1, 2, 3$ in the ionospheric model in (6.4) and leaving all other variables fixed.

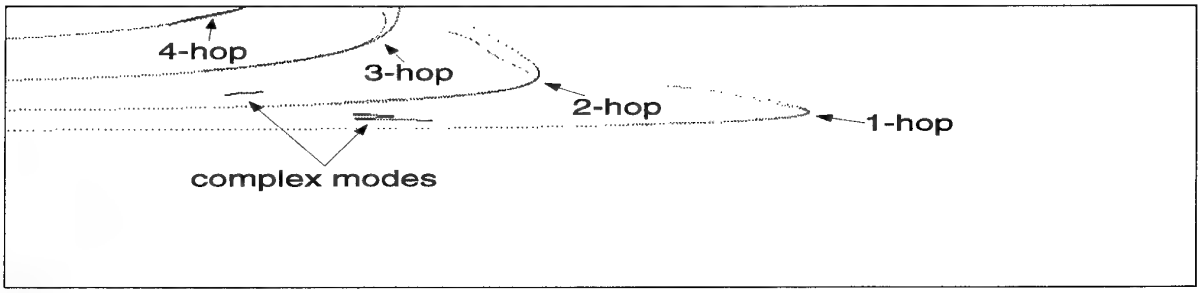


Figure 6.6: An example of cross-matching the ionospheric trace model against the fitted traces of an oblique ionogram. Even though the traces of the simplified ionospheric model and the trace fits do not correspond perfectly, there is enough to allow us to confidently identify the 1-hop, 2-hop, 3-hop and 4-hop traces. By elimination, all the remaining modes must be due to complex propagation paths.

most value of the fitted trace, $x_{\max}$. A one dimensional search through f_1 will be required to determine the junction frequency of the ionospheric trace model.

Once the initial guess has been determined for h_0, b, and f_s, the next step is to call an optimization procedure with a suitable objective function to refine the correspondence between the oblique trace model and the trace fit. Any minimization algorithm that does not require derivatives would be suitable for this optimization procedure. Each invocation of the objective function will select a number of points on the fitted trace, perform a one-dimensional search through f_1 to find the closest points on the ionospheric trace model, and then compute the distance between each pair. The value returned by the objective function will be the sum of the square of the distances between these pairs. The number and distribution of the points along the fitted trace will have to be determined by experimentation.

6.2.3 Cross-Matching the Traces

The process of cross-matching the predicted higher hops of the ionospheric model (6.4) against the fitted traces of the ionogram will be straight forward (figure 6.6). All that is required is to determine if there is a trace that lies under the predicted higher hops for much of its length. Note that the predicted traces are least accurate around the nose region, so emphasis should be placed on the lower ray when determining a match.

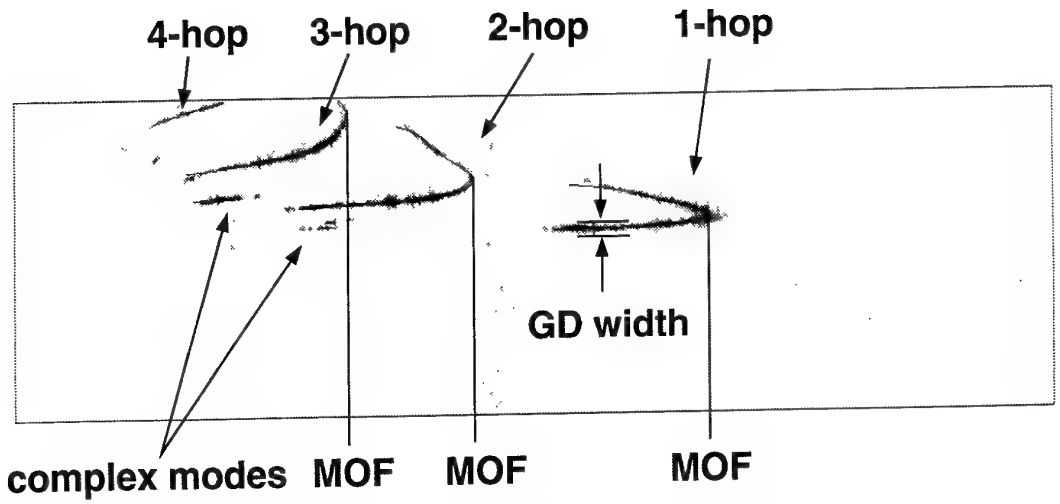


Figure 6.7: The scope of measurements that could be determined using the autoscaling system include the maximum observable frequency (MOF) which occurs at the nose of each of each simple propagation modes, and the group delay width of each of the traces. The complex modes are those modes left over after identifying the simple propagation modes.

6.3 Measurements

The full scope of the measurement has yet to be resolved, and will need to be more fully determined by user's requirements. None the less, the sort of measurements that could be determined by this system are illustrated in figure 6.7.

The principles that would be used to implement this stage include the following. Firstly, contiguous pixels in the ionogram will be matched against the mode-identified trace-fits, and this will give the pixels in ionograms that correspond to particular identifiable propagation modes. Figure 6.7 has the trace fits for all the significant and identifiable traces overlaid on the filtered ionogram to demonstrates this principle. Secondly, measurements will be guided by the polynomial fits. That is, the junction frequency of each of the simple modes occurs in the region where there is a zero-crossing in the first derivative of the fitting polynomial. Lastly, to cater for any discrepancy between the trace fit and the actual ionogram trace, all measurements will be taken from the actual ionogram data. For example, once the position of the junction has been determined on the fit, this information is used to search in this region of the original ionogram to find the actual junction frequency on the ionogram. This will eliminate any error induced by a mismatch between trace and its fitted polynomial.

Chapter 7

Assessment

In this chapter the performance of the autoscaling system is examined against 25 ionograms selected by a potential user to represent a range of observed phenomena. These ionograms are by no means statistically representative and were chosen because they highlight some of the unusual effects that are observed. Finally, this chapter finished with some comments on further work that is needs and a conclusion.

7.1 Assessment Results

The assessment ionograms at various stages of processing can be seen in figures 7.1–7.25. The trimmed ionograms (figures 7.1(d)–7.25(d)) were obtained from the thinned ionograms using an improved trimming algorithm [16] based on the proposed algorithm presented in section 3.6. Table 7.1 presents an assessment of the problems and indicates what areas need to be addressed to improve the performance of the autoscaling system as it stands.

It is apparent from these ionograms that the single greatest improvement to the system will be obtained by making the changes to the fitting stage suggested in section 4.6.2. The wildly oscillating trace fits that are characteristic of failed fitting (because of either an inappropriate model or limitations of the ODR fitting routine (section 4.5)) have obscured many of the successful fits and consequently may make the performance of the system seem worse than it is. Implementing these changes would eliminate all the problems highlighted in table 7.1 by a "2", "3", or "5". Using the improved trimming algorithm of [16] based on section 3.6 has eliminated almost all of the problems associated with high adjacency in ionograms and consequently the erroneous connectivity between traces of differing propagation modes that results. Implementing a branch selection scheme according to section 3.7 (which was achieved by hand here when a particular ionogram required separation of o- and x-rays) would deal with all the cases highlighted by a "1" in table 7.1. It is hard to say much about the performance of the merging routine until the other problems are fixed along with improving the estimate of variance of the residuals (section 5.3). If this estimate was upgraded, then I would expect that most, if not all, of the problems of missing a potential merge (indicated by a "6") would be eliminated. At least there is no evidence of the merging routine doing damage to the traces by erroneously connecting two traces of differing propagation modes. In this collection of ionograms, there are three that contain traces that have been broken into more than two segments (indicated by a "7"). This provides evidence that sometimes, at least, it is necessary to consider merges of more than two components.

There is one ionogram in the collection that requires special treatment for a phenomena called "Spread-F" that causes wide diffuse traces to form in the ionogram that are so diffuse that nothing useful can be determined from these traces (figure 7.10 and indicated by an "8" in table 7.1). When this is present in a trace, it is necessary to flag it and ensure that it is not accidentally passed to the existing stages of the autoscaling system. Regions of such phenomena could be recognized from clusters of vertices in the piecewise-linear approximation to the thinned trace structure, but this idea will need some work. These regions were deleted by hand in assessment ionogram 10 between figures 7.10(d)–(e) using a pixel editing program.

Finally, it is worth pointing out that none of the problems highlighted by these ionograms were

unexpected and their solutions have been presented in earlier chapters, except for the problem associated with spread-F phenomena.

7.2 Further Work

I have already pointed out a few deficiencies of the existing system that need to be addressed. The first, that of the need for an improved trimming algorithm, has already been solved [16] while this report was being written. For the others including branch selection, x- and o-ray separation, fitting model, fitting routine, and residual variance estimation I have indicated the improvements that need to be made. I don't foresee these to be more difficult than trimming to solve conclusively, and some of them (*eg.*, a new fitting model) will be a great deal easier to solve. The idea for detecting spread-F phenomena in thinned-trimmed traces will need further thought, but this is a problem that aflicts a minority of LLISP ionograms. The tracking and mode identification stages need to be implemented as presented in chapter 6. This will not be difficult because both of these stages are fairly clear-cut optimization problems as I have defined them. With these improvements in place, the autoscaling system will be a fully functional solution to the autoscaling problem.

One issue I have not dealt with in this report is how confidence measures can be obtained for the results the system produces. These measures are highly desirable, because it would allow the system to recognize and report when an ionogram trace falls outside those that it is capable of handling. The key to this is in the fitting stage. Because fitting is integral to the whole philosophy of this autoscaling system, when it fails for a trace then scaling will fail for this trace also (and the reverse is probably true too). Consequently, detecting fitting failure will allow us to determine confidence measures. We already know that the failure mode for fitting is for the trace fit to oscillate wildly backwards and forwards across the region of the trace. Therefore, the basis of a confidence measure is to determine when there are rapid sign changes in the derivative of the trace fit.

7.3 Conclusion

In conclusion, I have presented a system for autoscaling oblique ionograms that employs a number of successive steps divided up into three overall stages of filtering, feature extraction and measurement. The steps involved in feature extraction include thinning, trimming, branch selection, fitting and merging to reduce each trace of a particular propagation mode to a symbolic form. The measurement stage includes a tracking step to account for missing traces between successive ionograms, a mode identification stage and the final measurement step. When all the suggested improvements are in place, this system will offer a solution to the autoscaling problem across a broad range of oblique ionograms.

Table 7.1: Performance of Processing Stages on the Assessment Ionograms

Ionogram	Processing Stage					
	Initial Guess	Thinning	Trimming	Branch Selection	Fitting	Merging
1	√	√	√	1	2,3	√
2	√	√	√	√	2	√
3	√	4 (2-hop)	√	1	2	5
4	√	√	√	√	√	3 (2-hop), 6
5	√	√	√	√	2,3	6
6	√	√	√	1	√	6
7	√	√	√	1	2	√
8	√	√	√	1	2,3	√
9	√	√	√	1	2,3	6
10	√	√	√	1,8	2,3	√
11	√	√	√	1	√	3 (3-hop), 6 (4-hop), 7 (2-hop)
12	√	√	√	√	3	3 (1-hop, 3-hop)
13	√	√	√	1	2,3	√
14	√	√	√	1	2,3	6
15	√	√	√	1	2,3	6
16	√	√	√	√	√	√
17	√	√	√	1	2,3	6,7
18	√	√	√	1	2,3	√
19	√	√	√	1	2,3	√
20	√	√	√	1	√	6
21	√	√	√	1	√	3 (1-hop)
22	√	√	√	√	2,3	5
23	√	√	√	√	√	5, 6 (2-hop)
24	√	√	√	√	√	6
25	√	√	√	1	√	6, 7

Notes:
1 The presence of o- and x-rays requires special treatment.
2 The model used for fitting is inappropriate for some traces.
3 The optimization routine used for fitting needs improvement for some traces.
4 Thinning could be improved on some of these traces.
5 An inappropriate model used in fitting has meant that some traces were not merged that should be. This is a deficiency of the fitting stage, not of merging.
6 Some traces were not merged that could be. Better estimates for the variance of the residuals is required.
7 Contains traces that were broken into more than two fragments.
8 Special treatment was required for the presence of Spread-F phenomena which can cause problems for branch selection if it is not dealt with.

Figure 7.1: Assessment Ionogram 1.

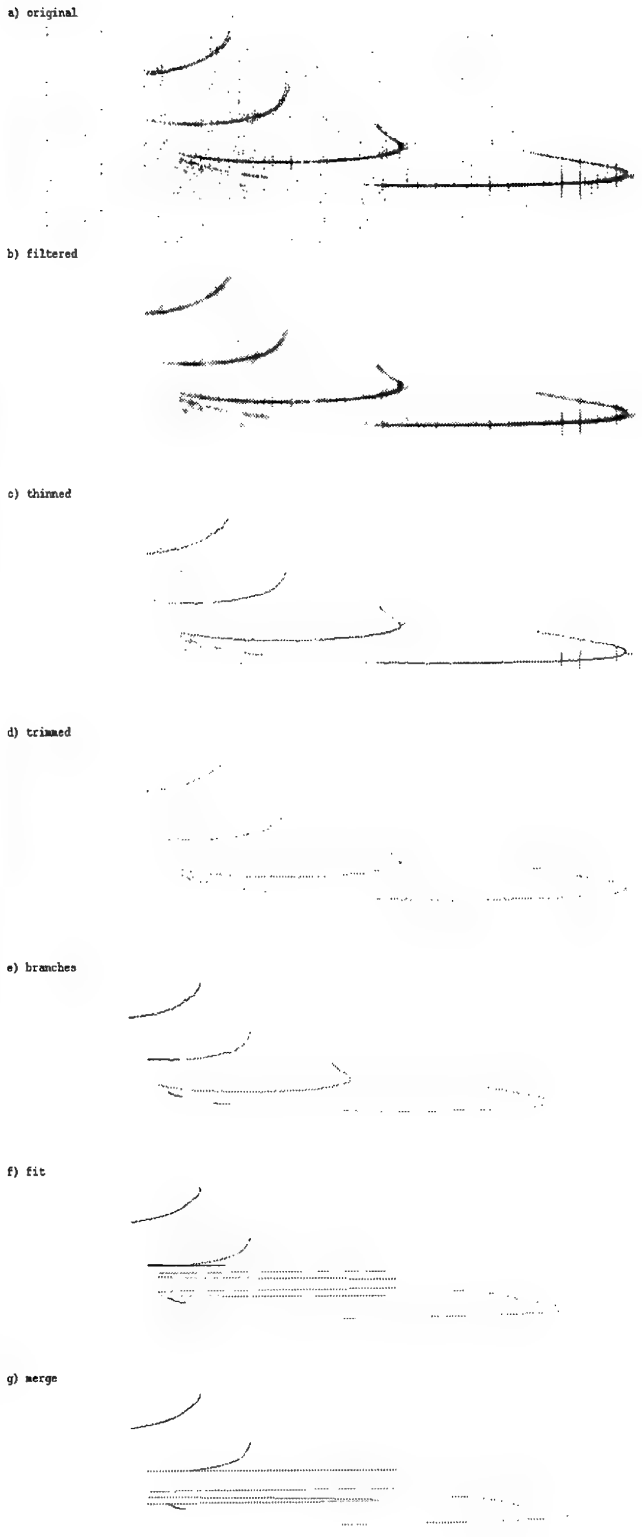


Figure 7.2: Assessment Ionogram 2.

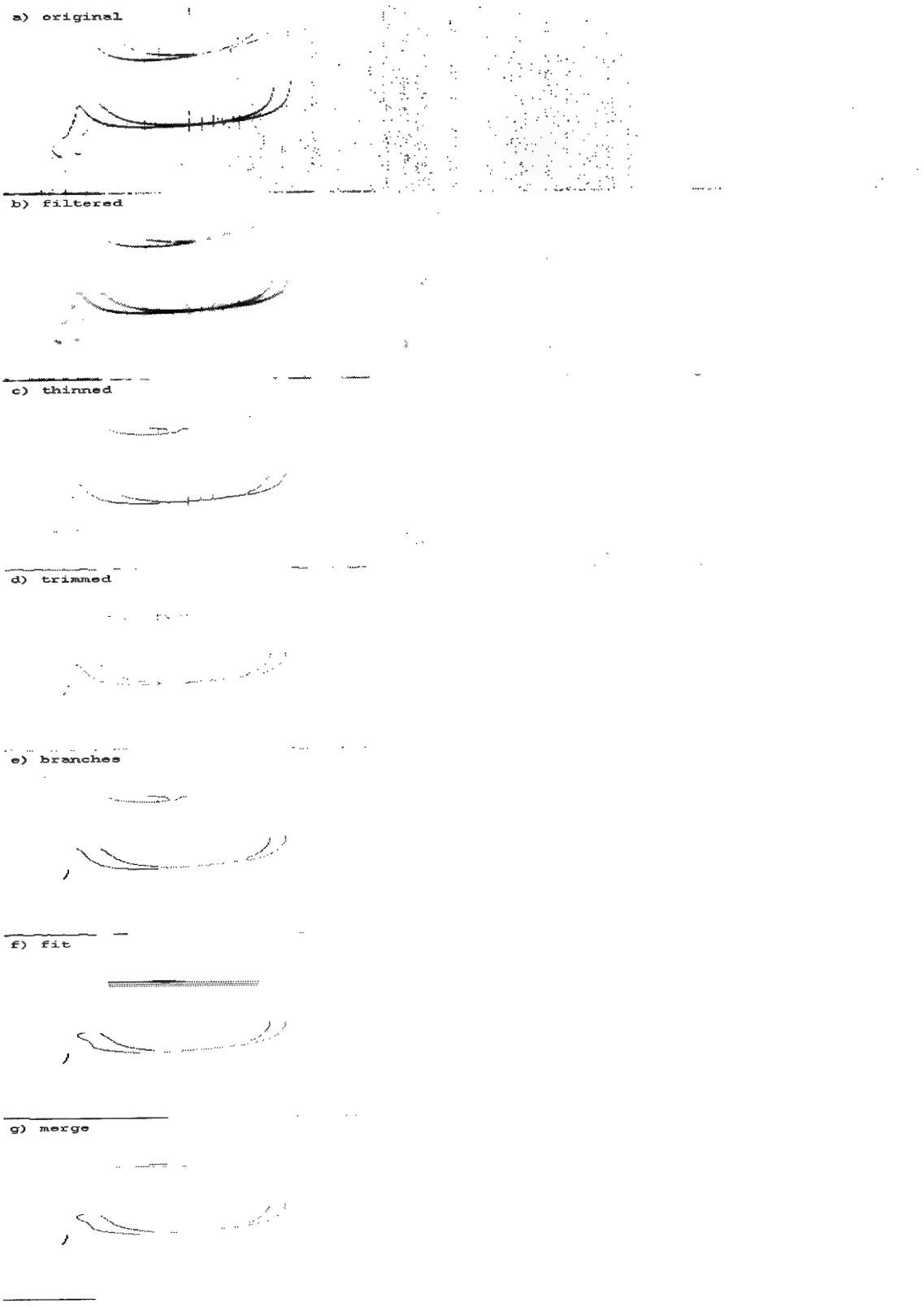


Figure 7.3: Assessment Ionogram 3.

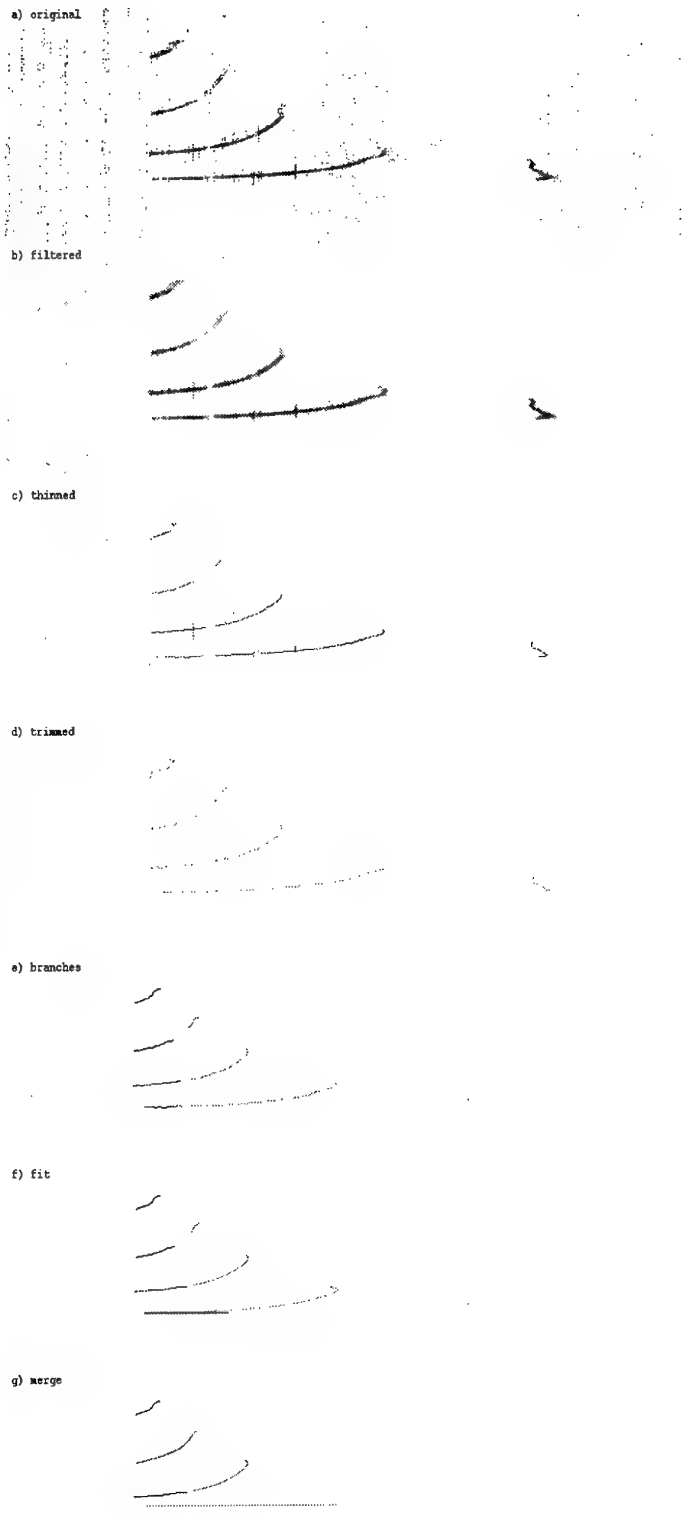


Figure 7.4: Assessment Ionogram 4.

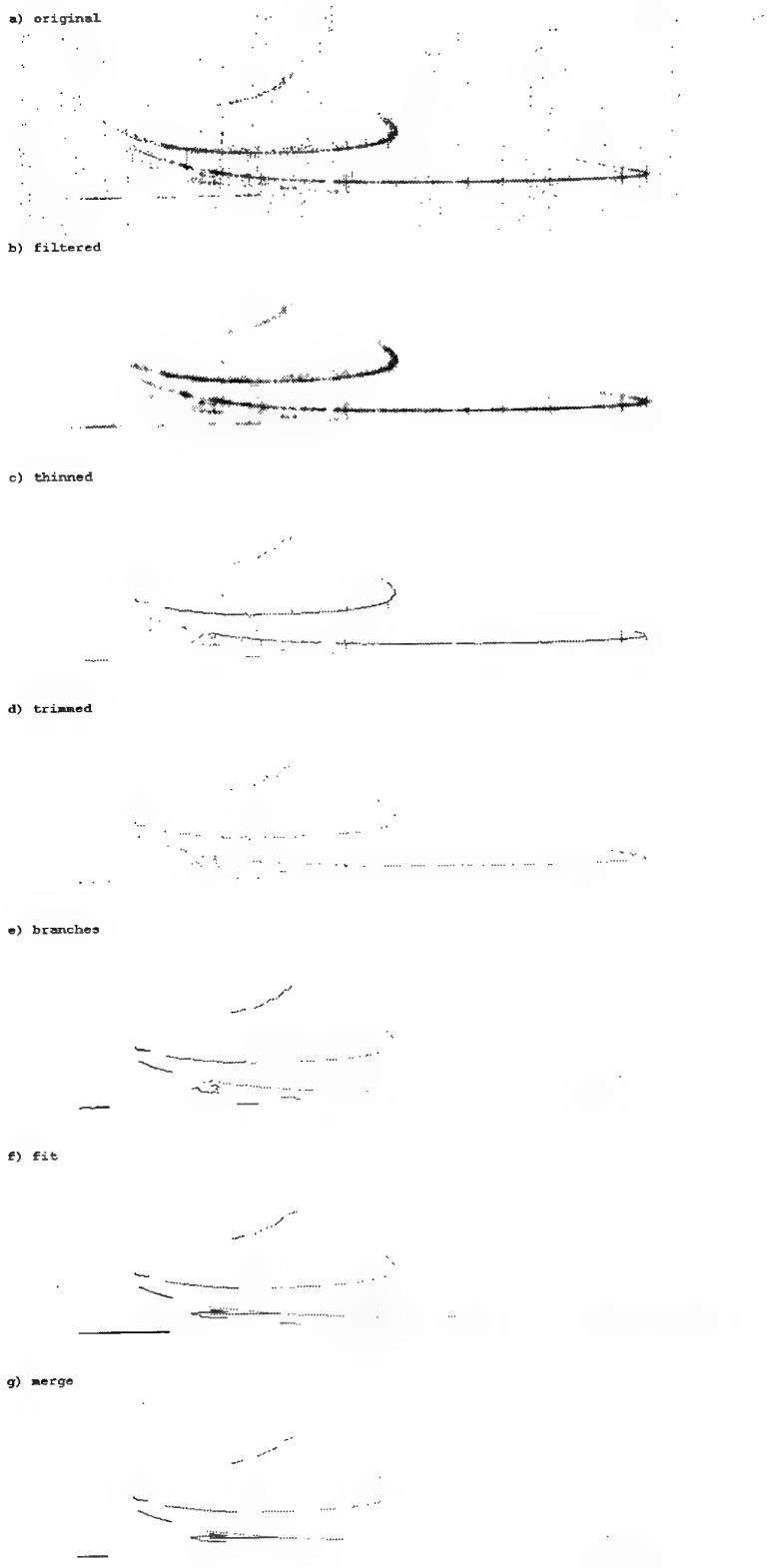


Figure 7.5: Assessment Ionogram 5.

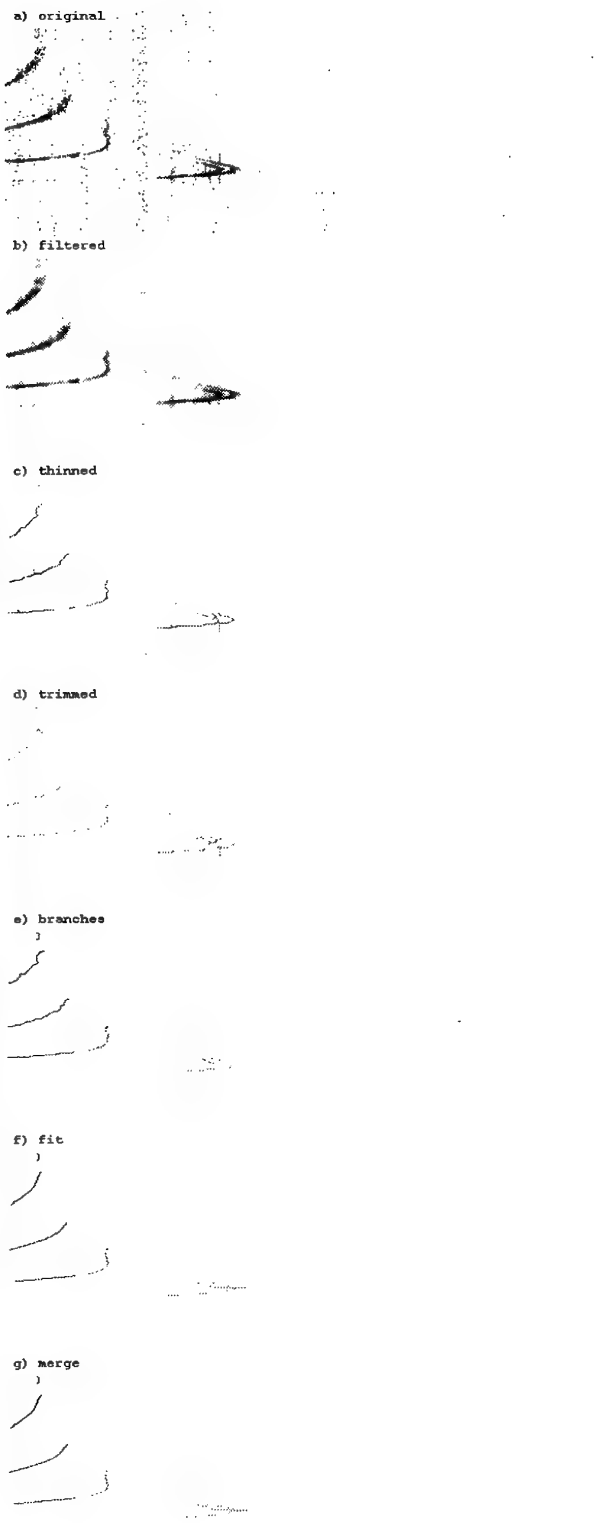


Figure 7.6: Assessment Ionogram 6.

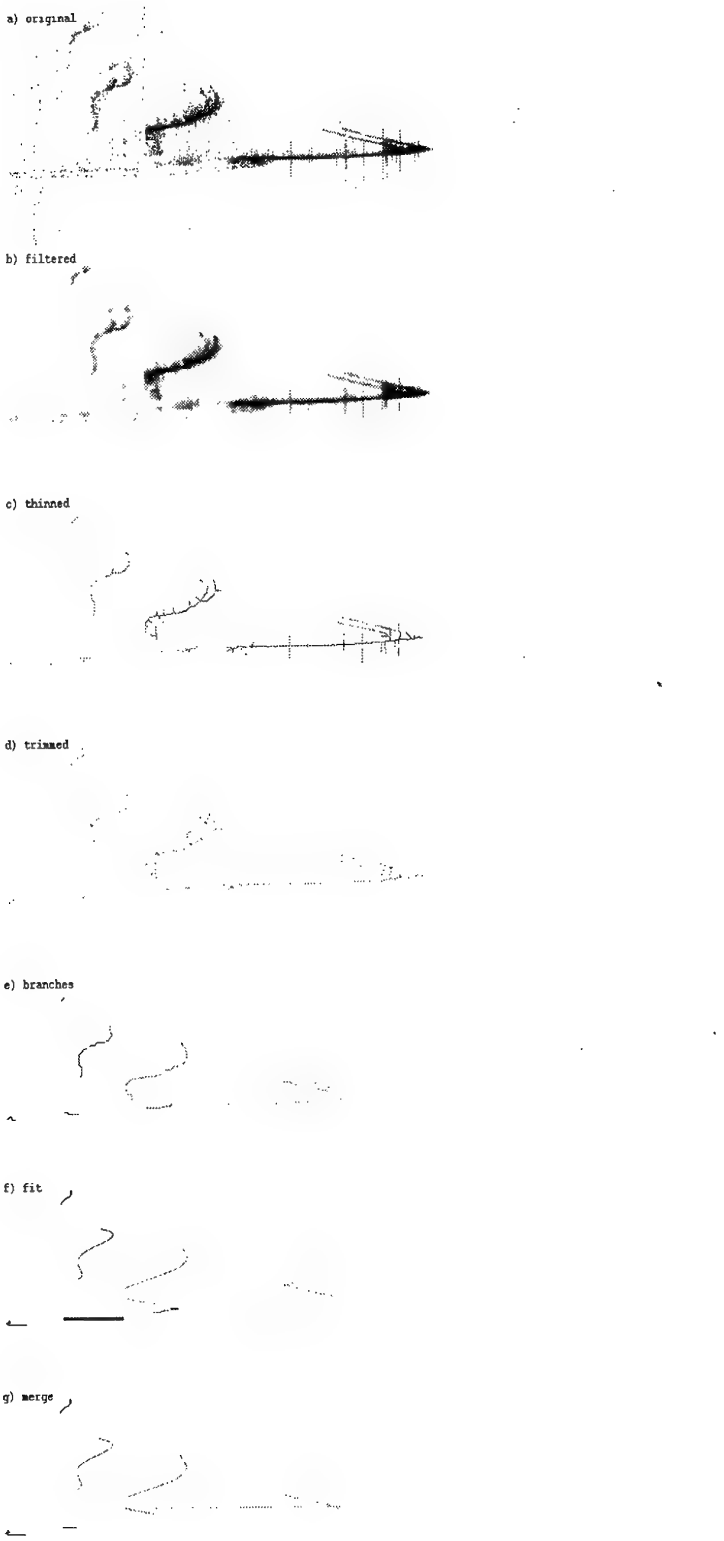


Figure 7.7: Assessment Ionogram 7.

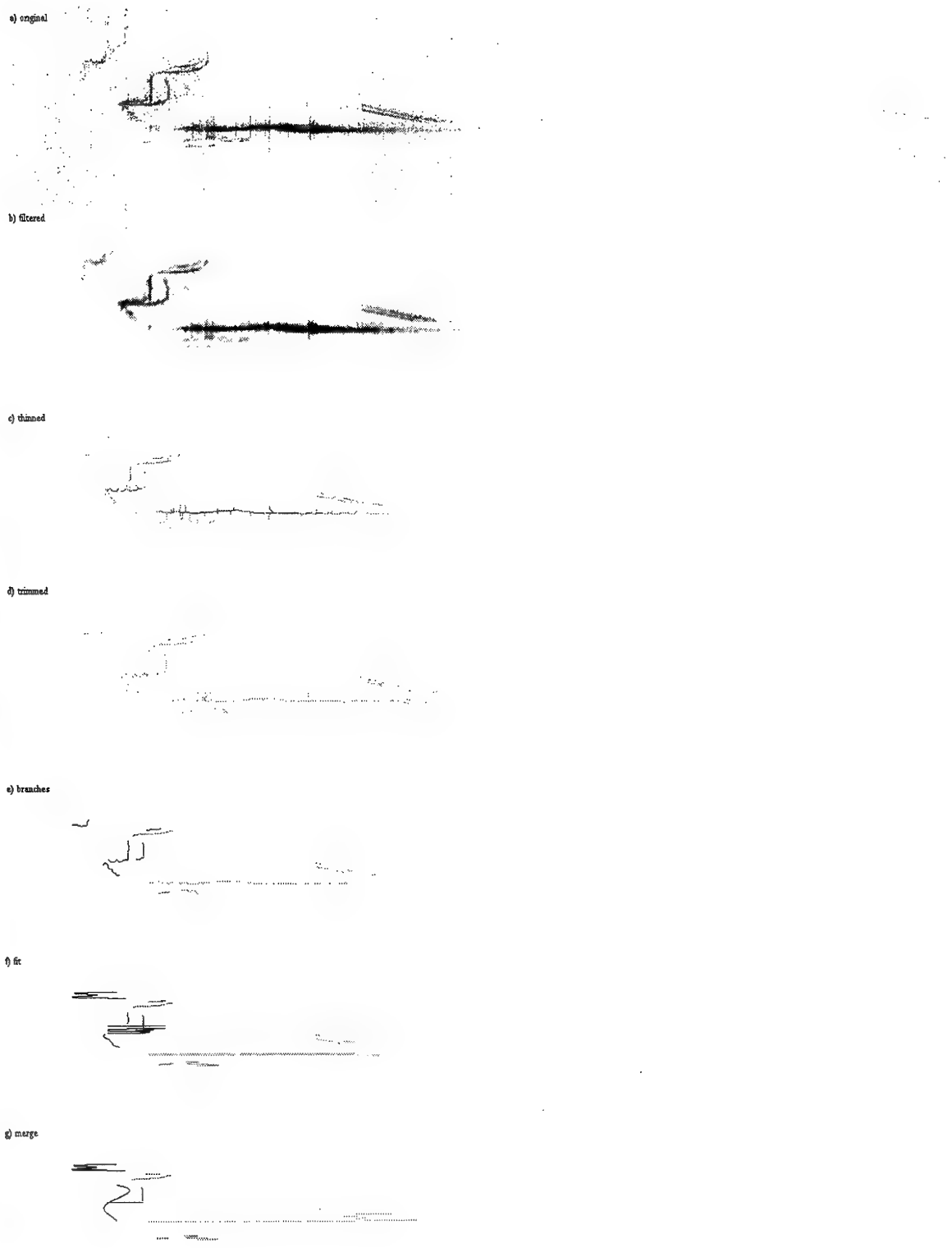


Figure 7.8: Assessment Ionogram 8.

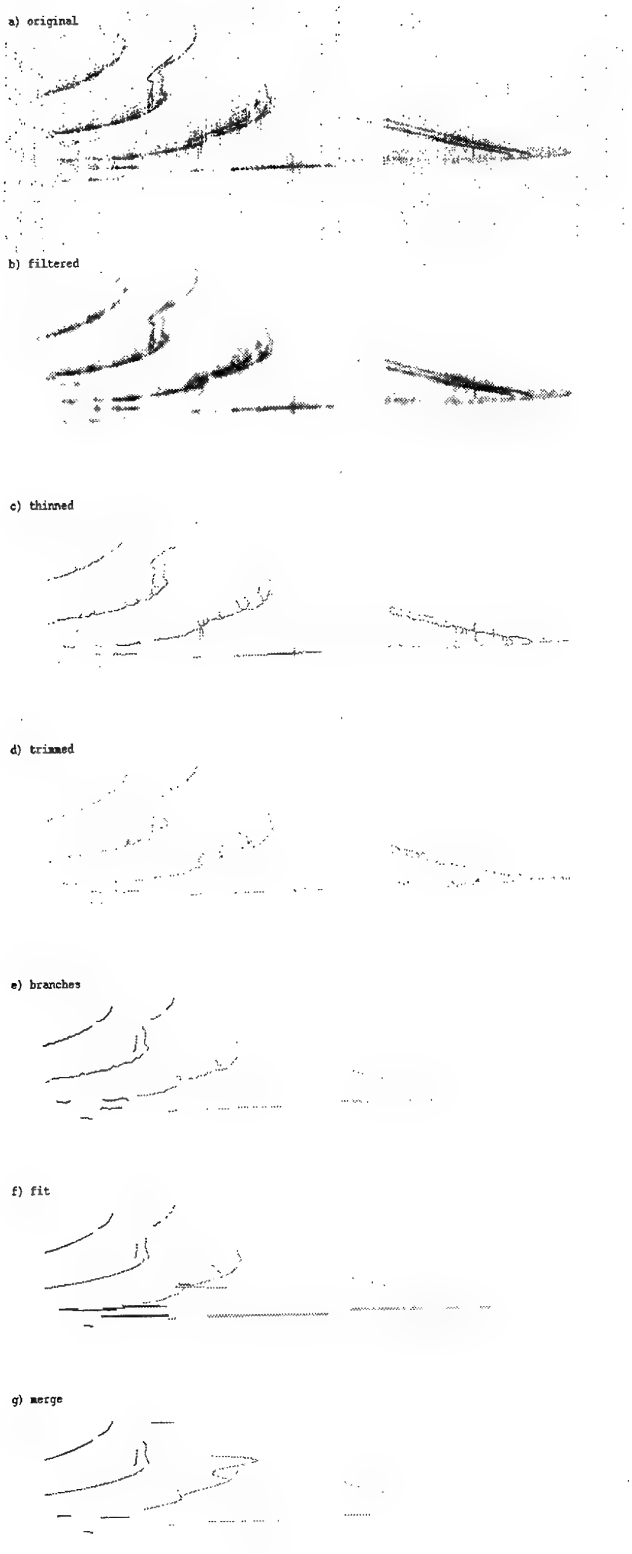


Figure 7.9: Assessment Ionogram 9.

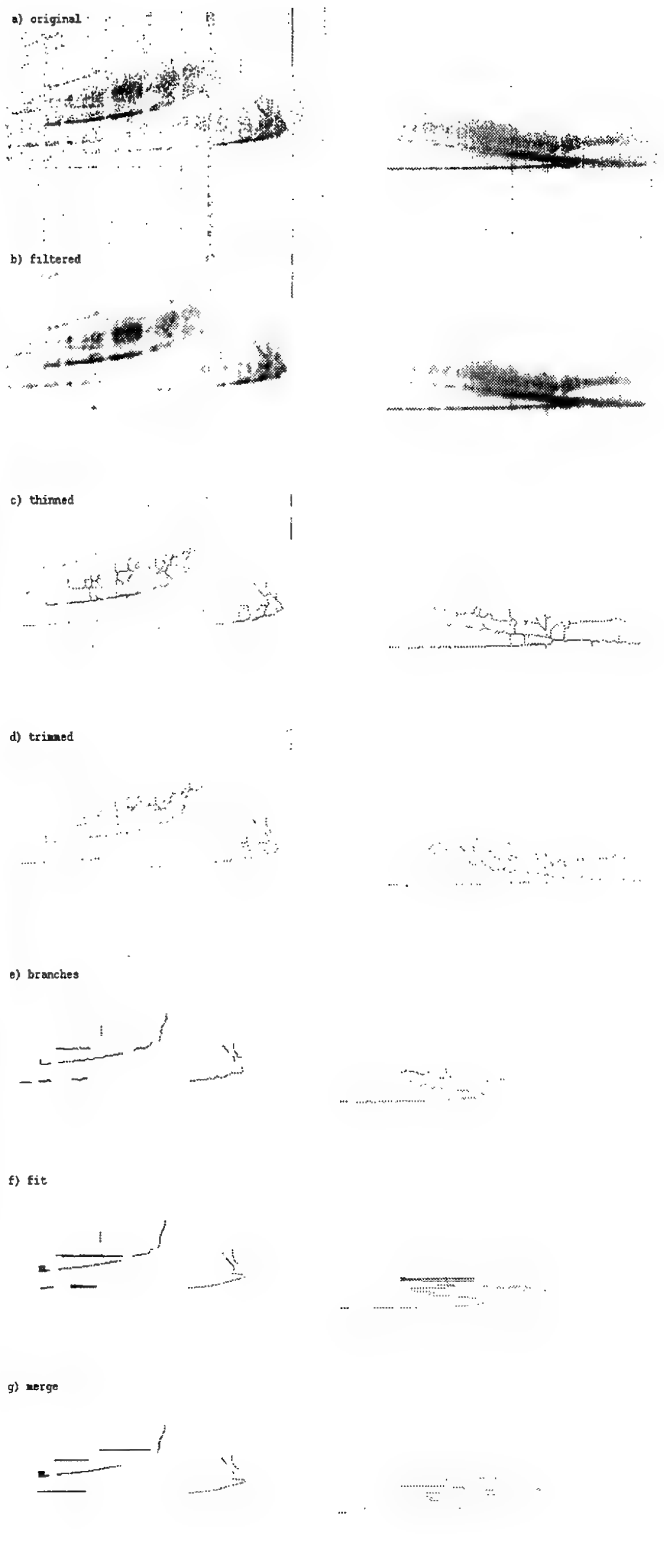


Figure 7.10: Assessment Ionogram 10.

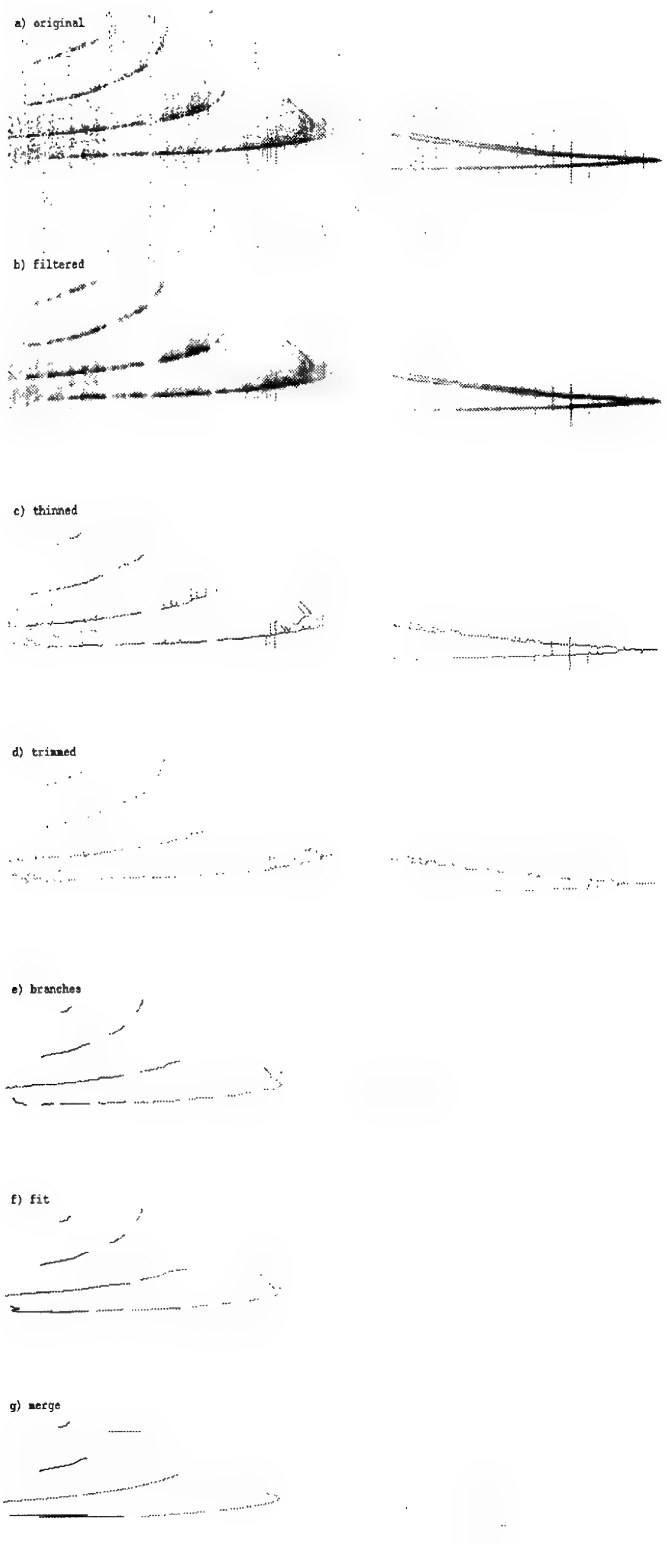


Figure 7.11: Assessment Ionogram 11.

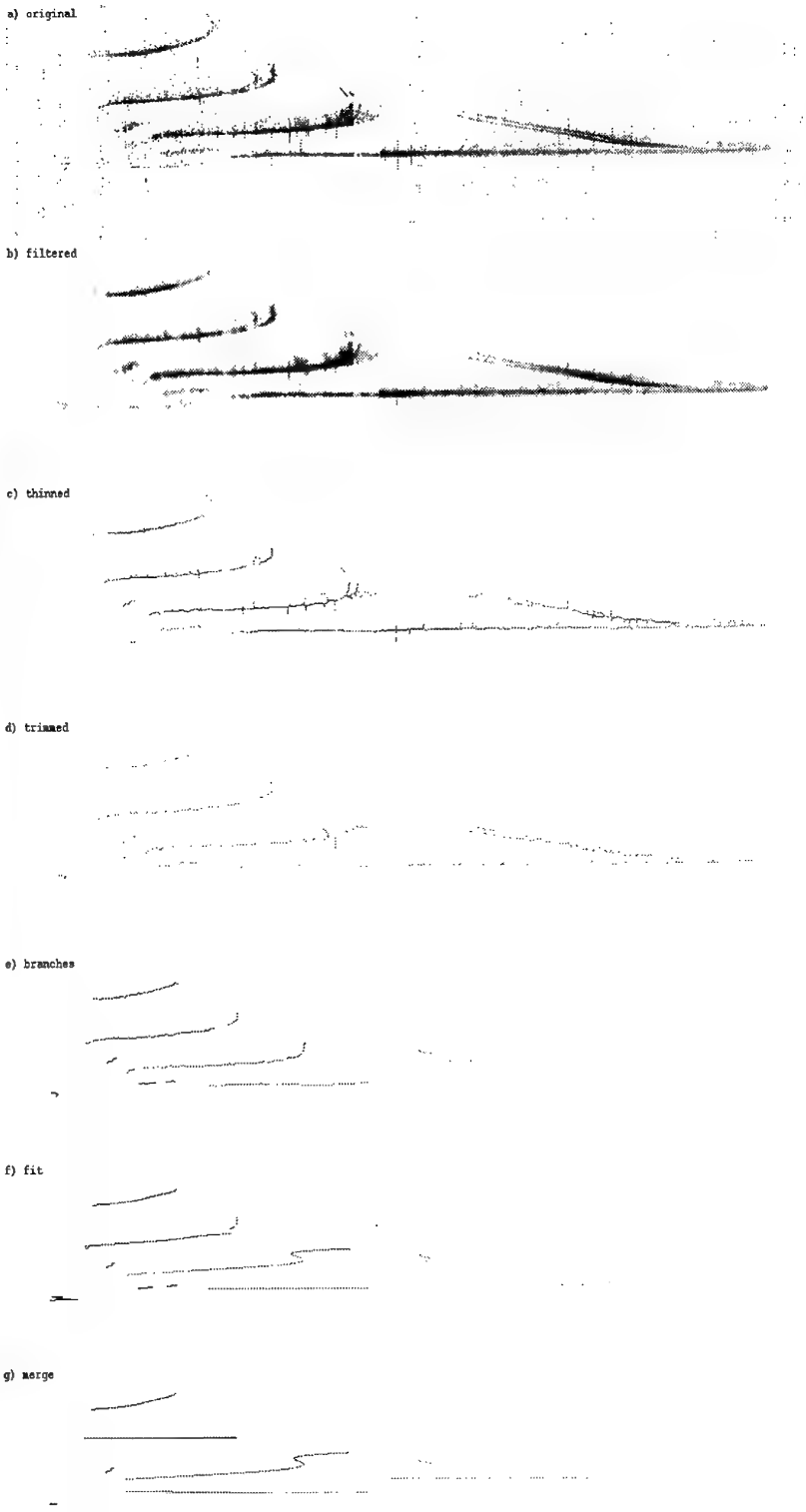


Figure 7.12: Assessment Ionogram 12.

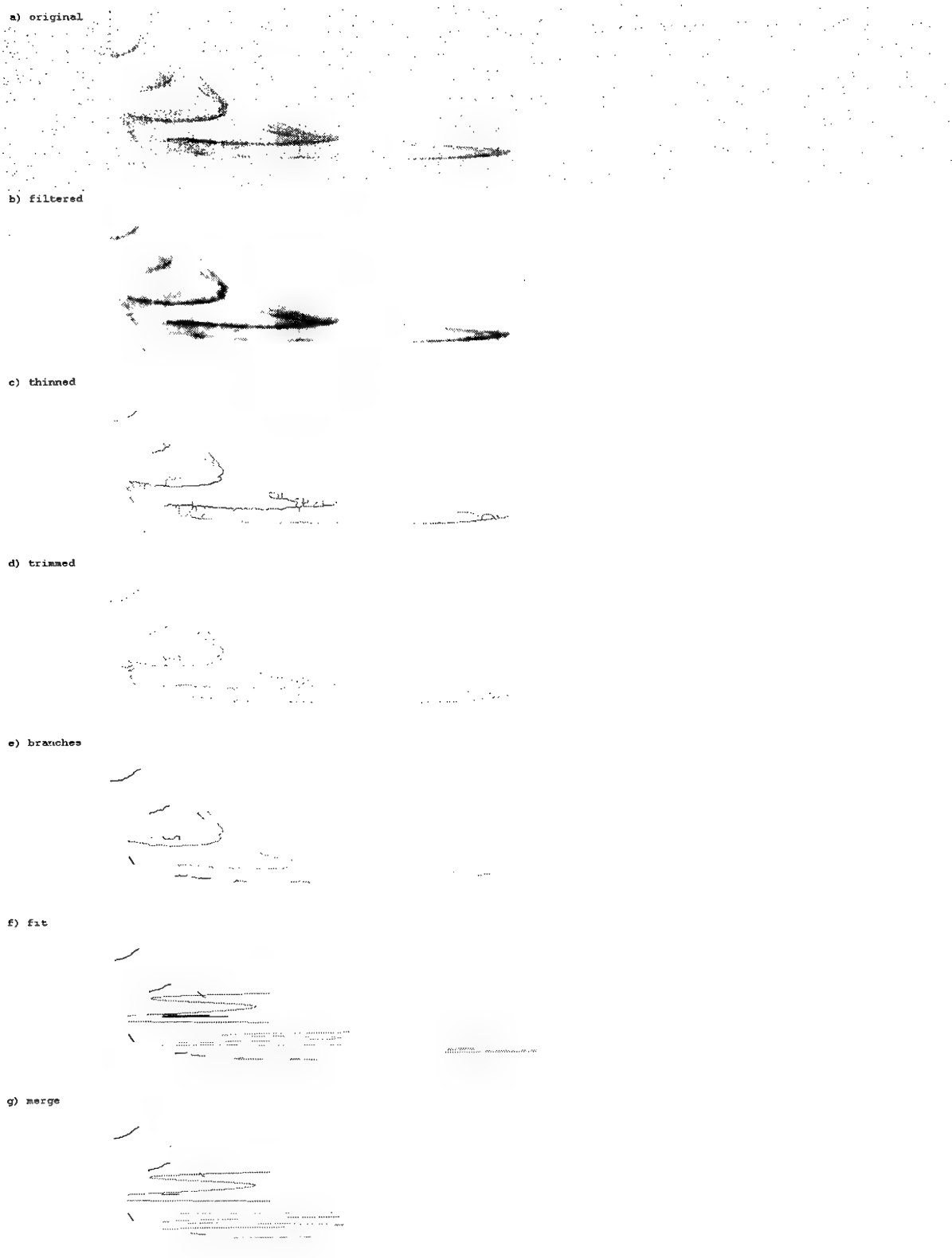


Figure 7.13: Assessment Ionogram 13.

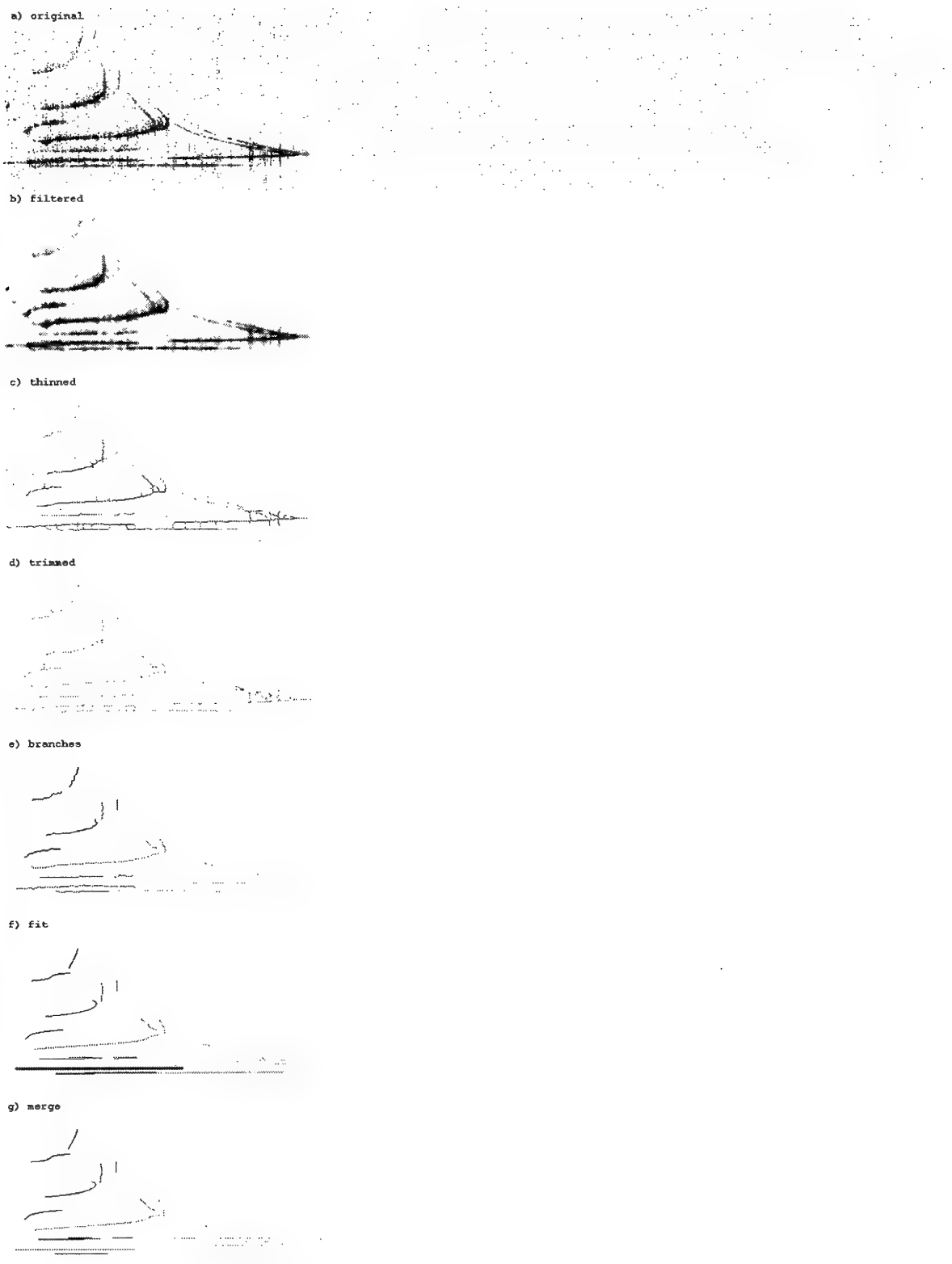


Figure 7.14: Assessment Ionogram 14.

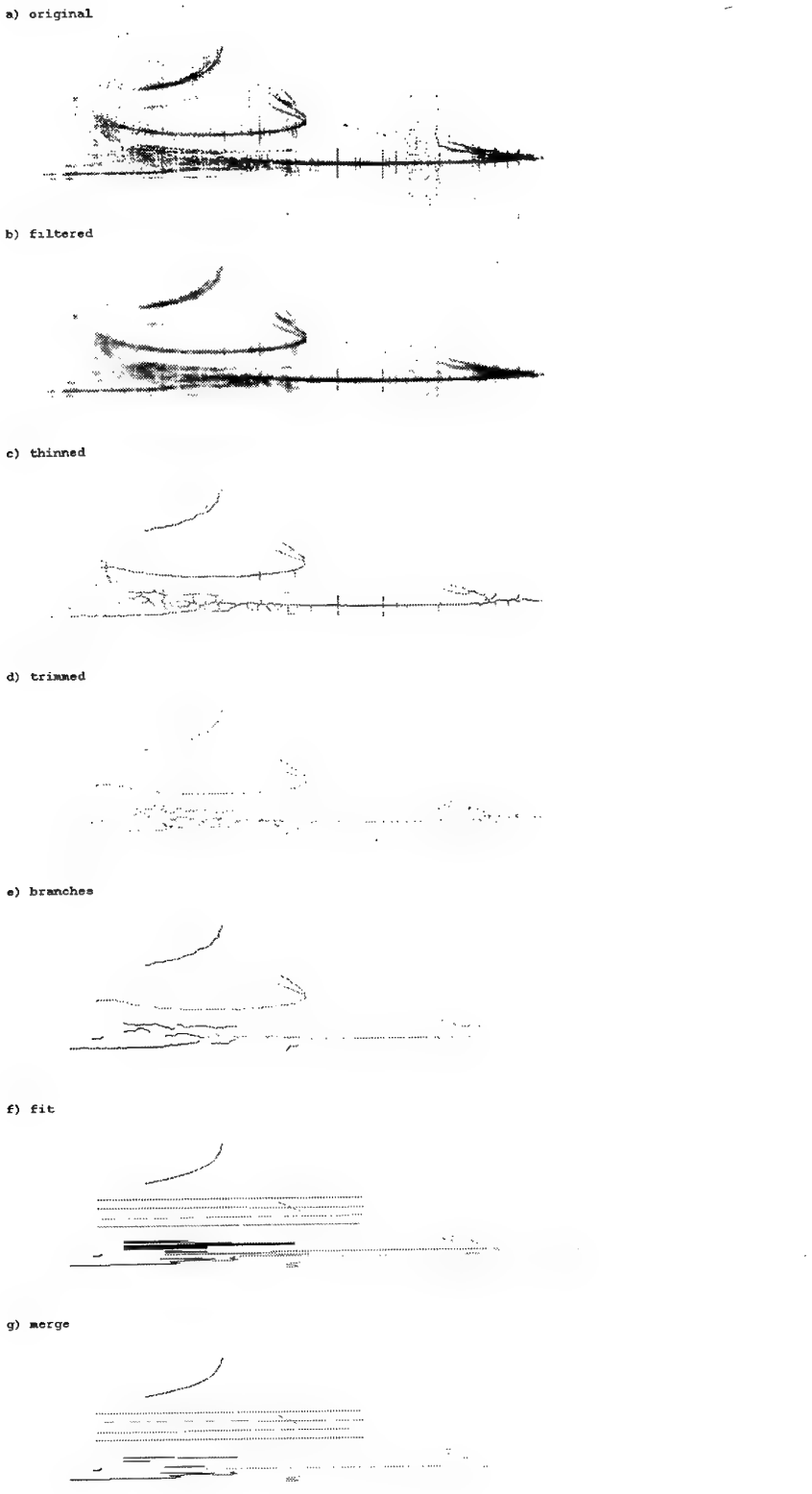


Figure 7.15: Assessment Ionogram 15.

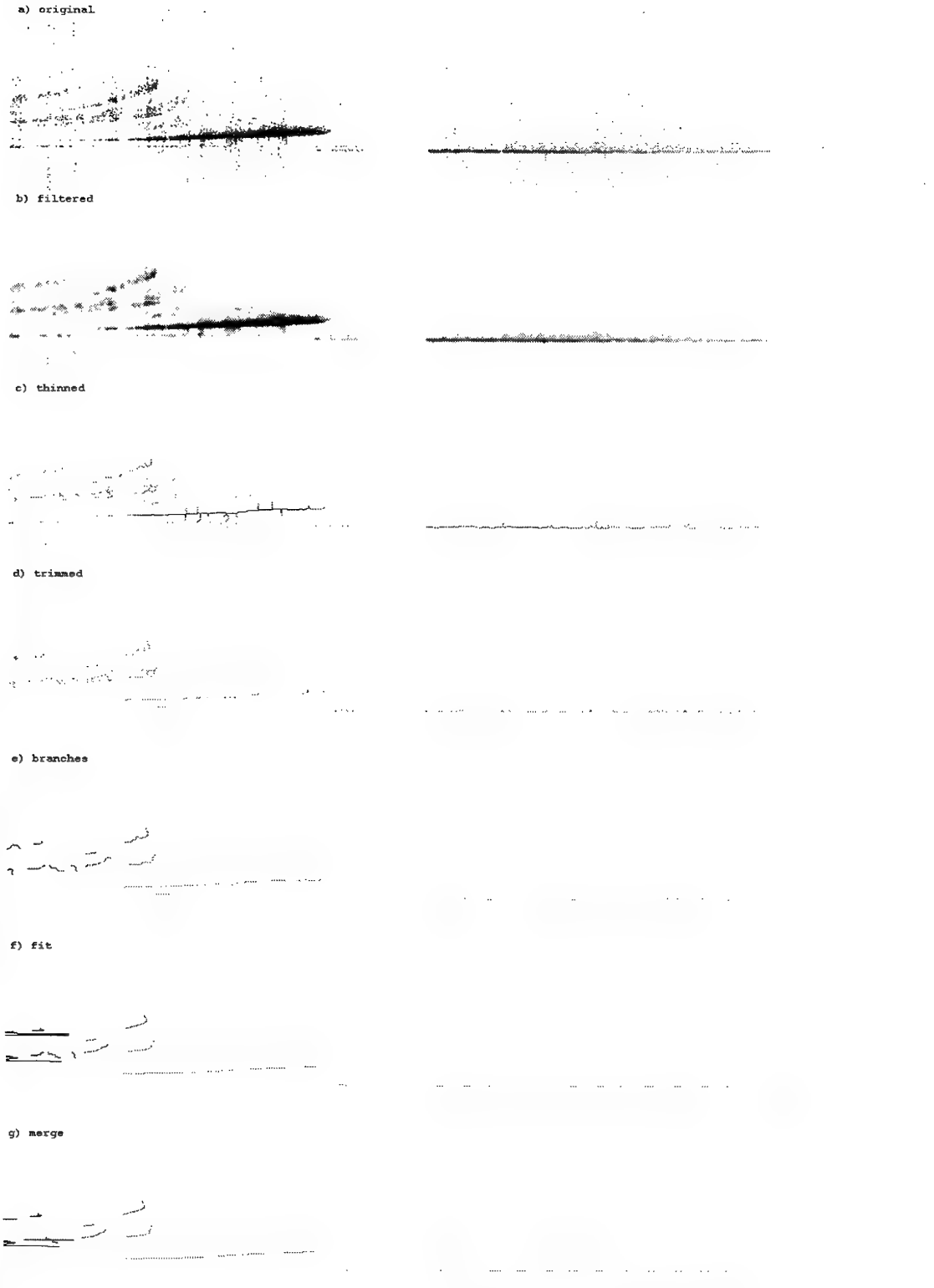


Figure 7.16: Assessment Ionogram 16.

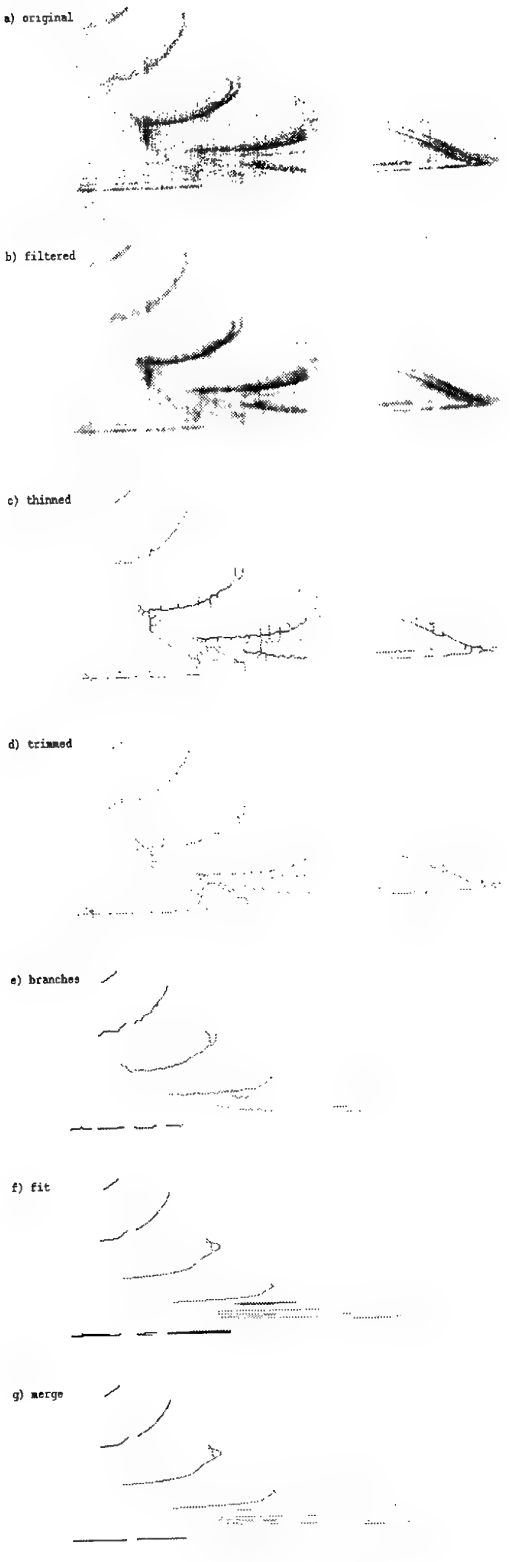


Figure 7.17: Assessment Ionogram 17.

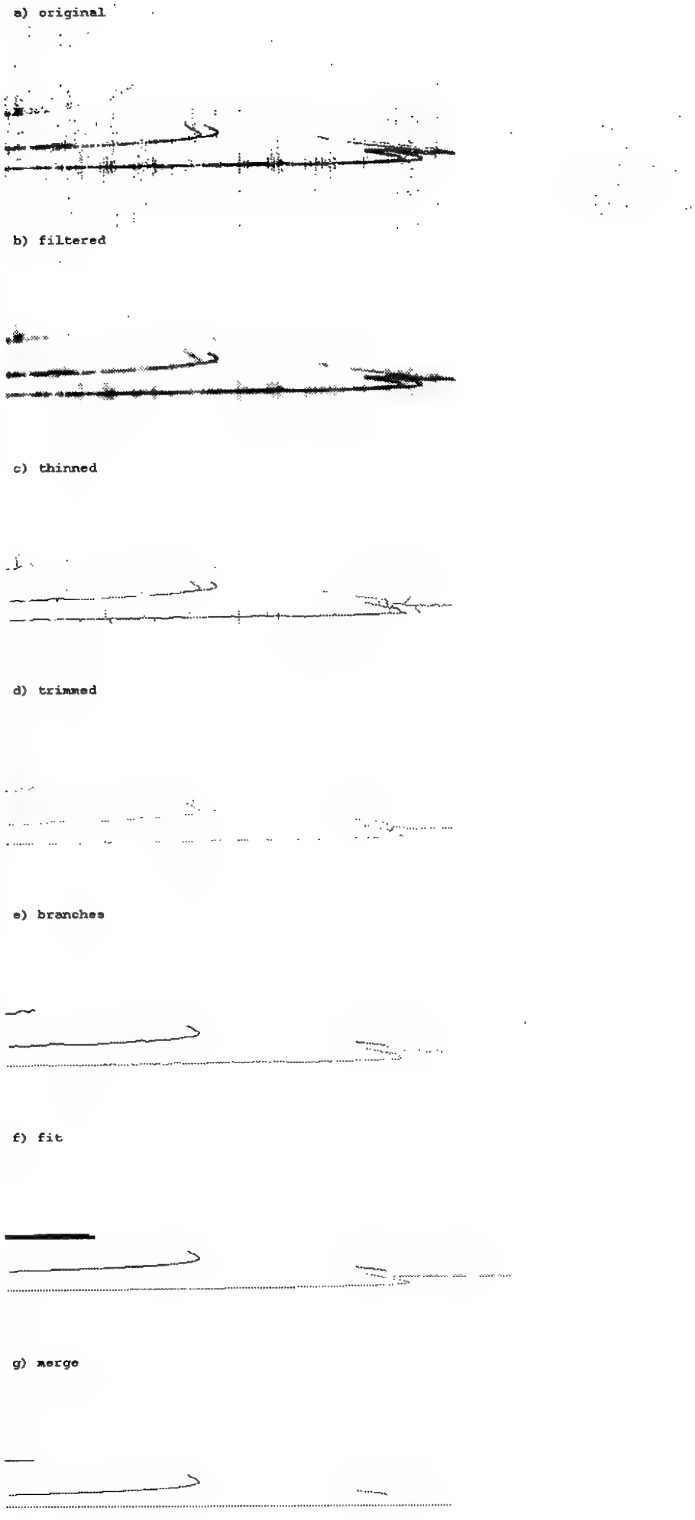


Figure 7.18: Assessment Ionogram 18.

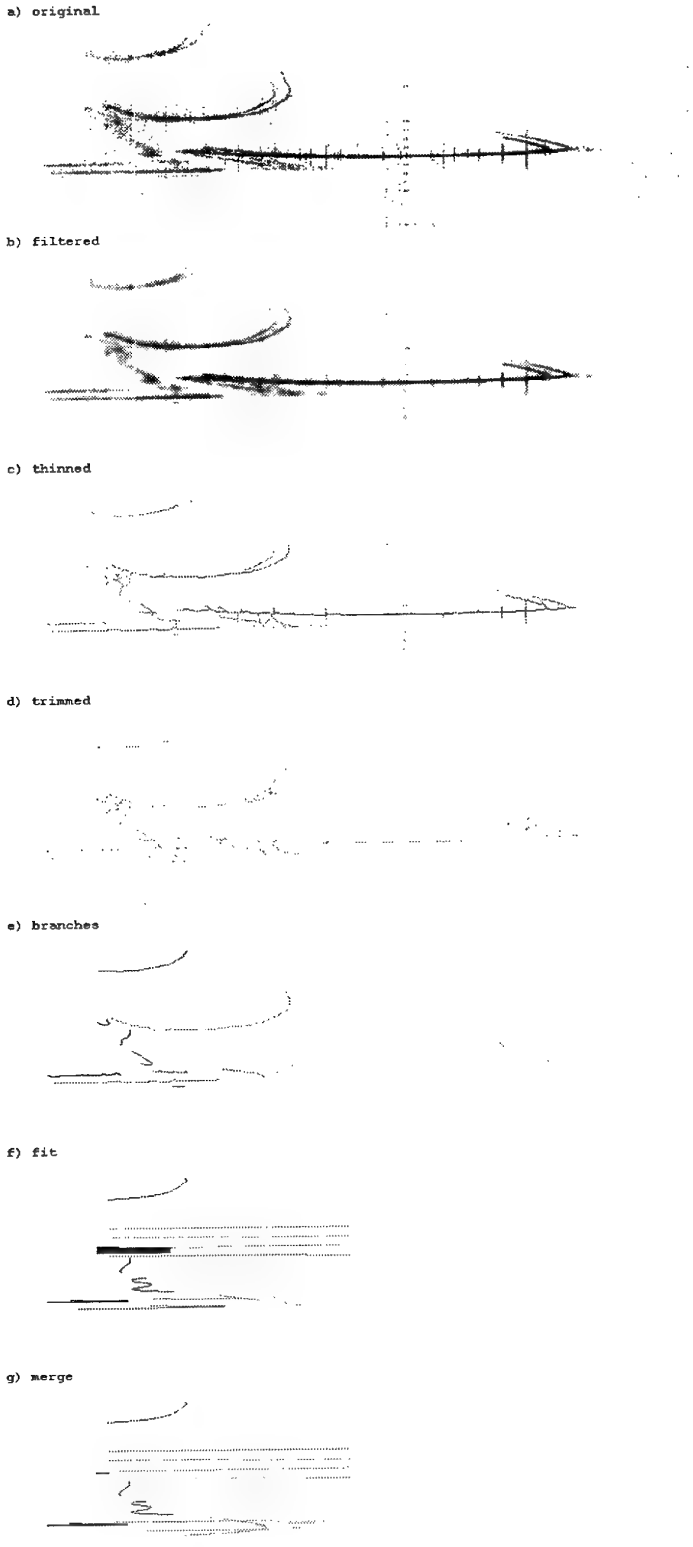


Figure 7.19: Assessment Ionogram 19.

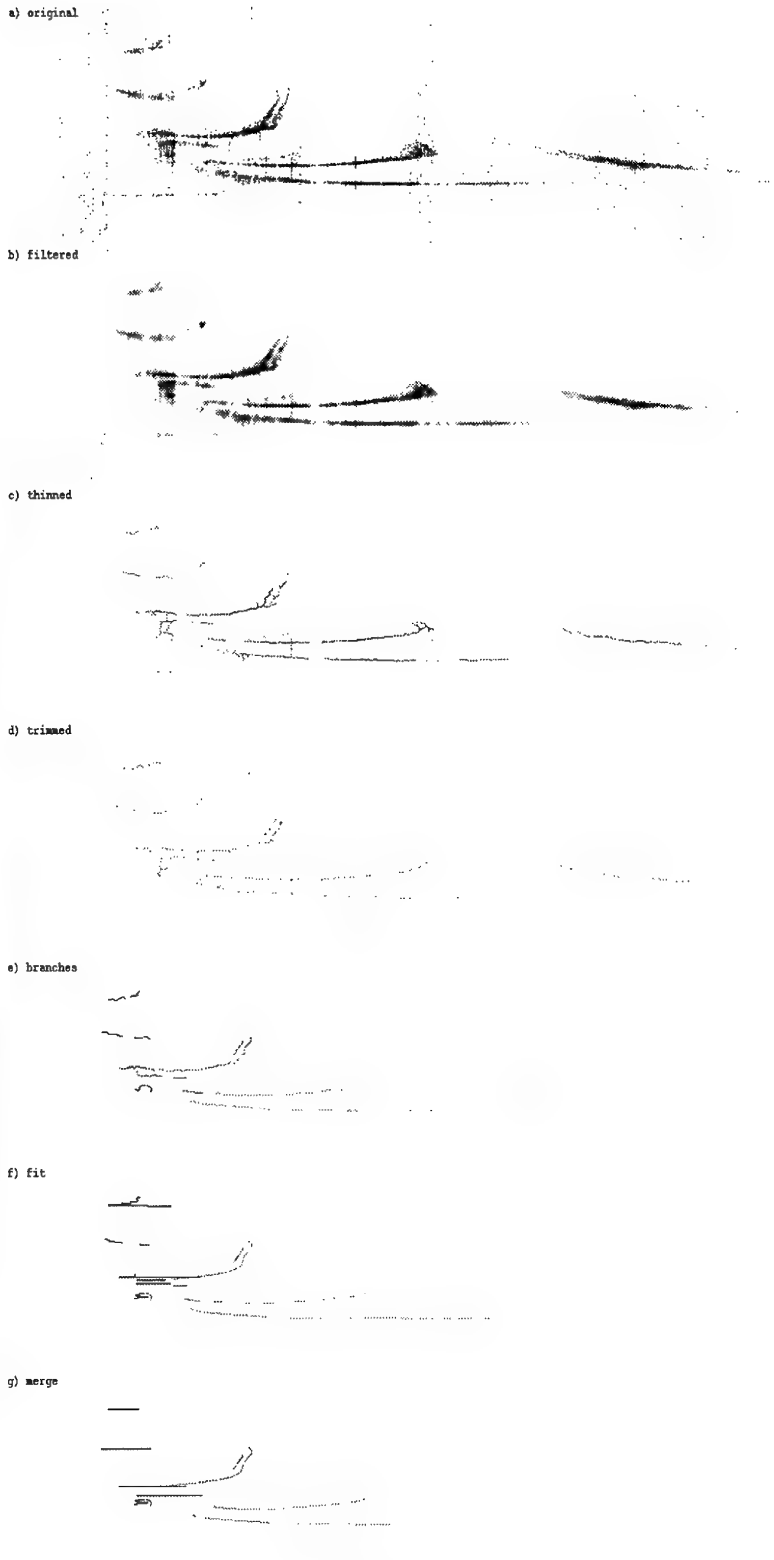


Figure 7.20: Assessment Ionogram 20.

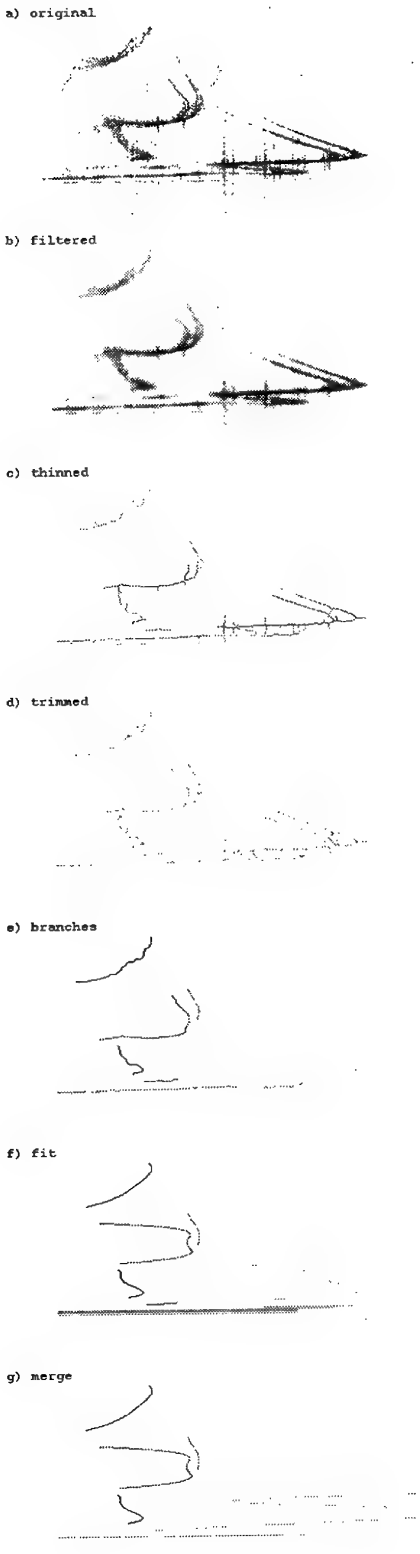


Figure 7.21: Assessment Ionogram 21.

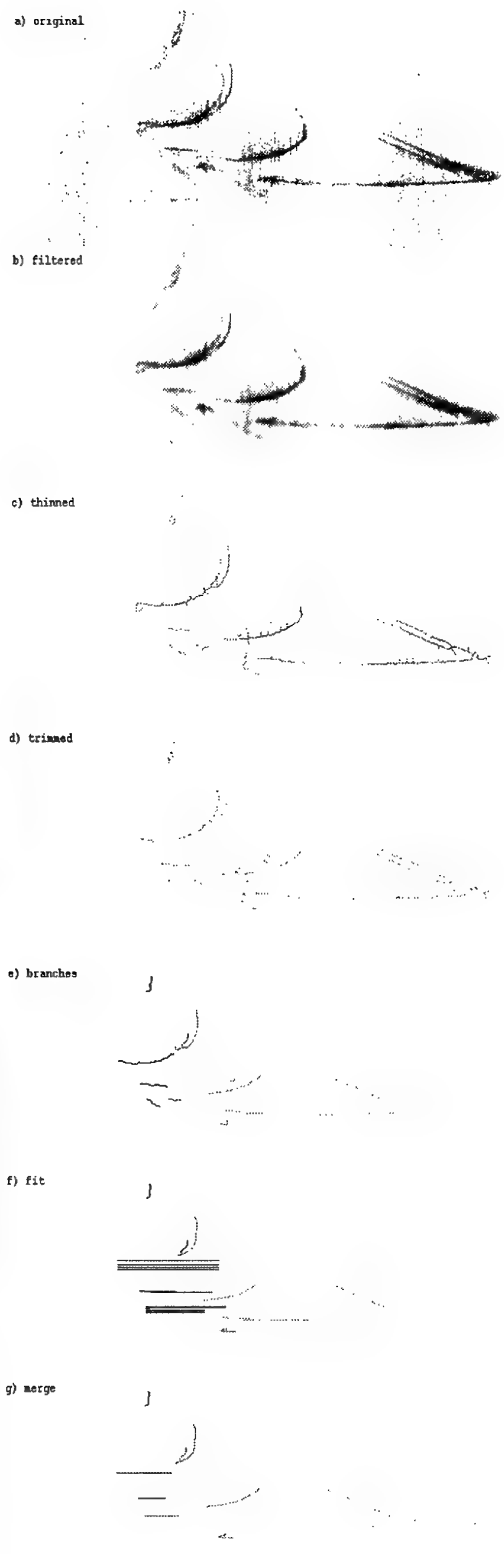


Figure 7.22: Assessment Ionogram 22.

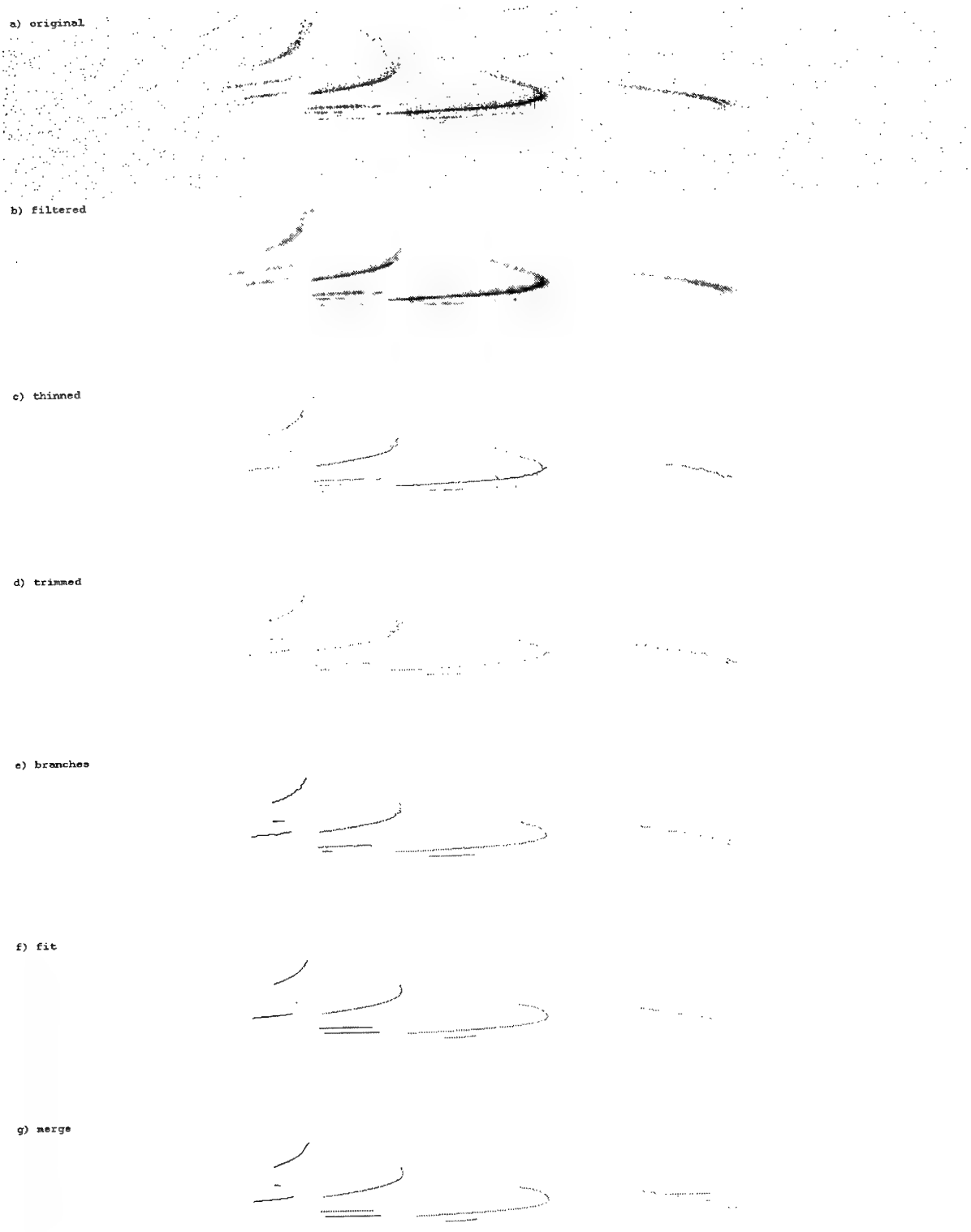


Figure 7.23: Assessment Ionogram 23.

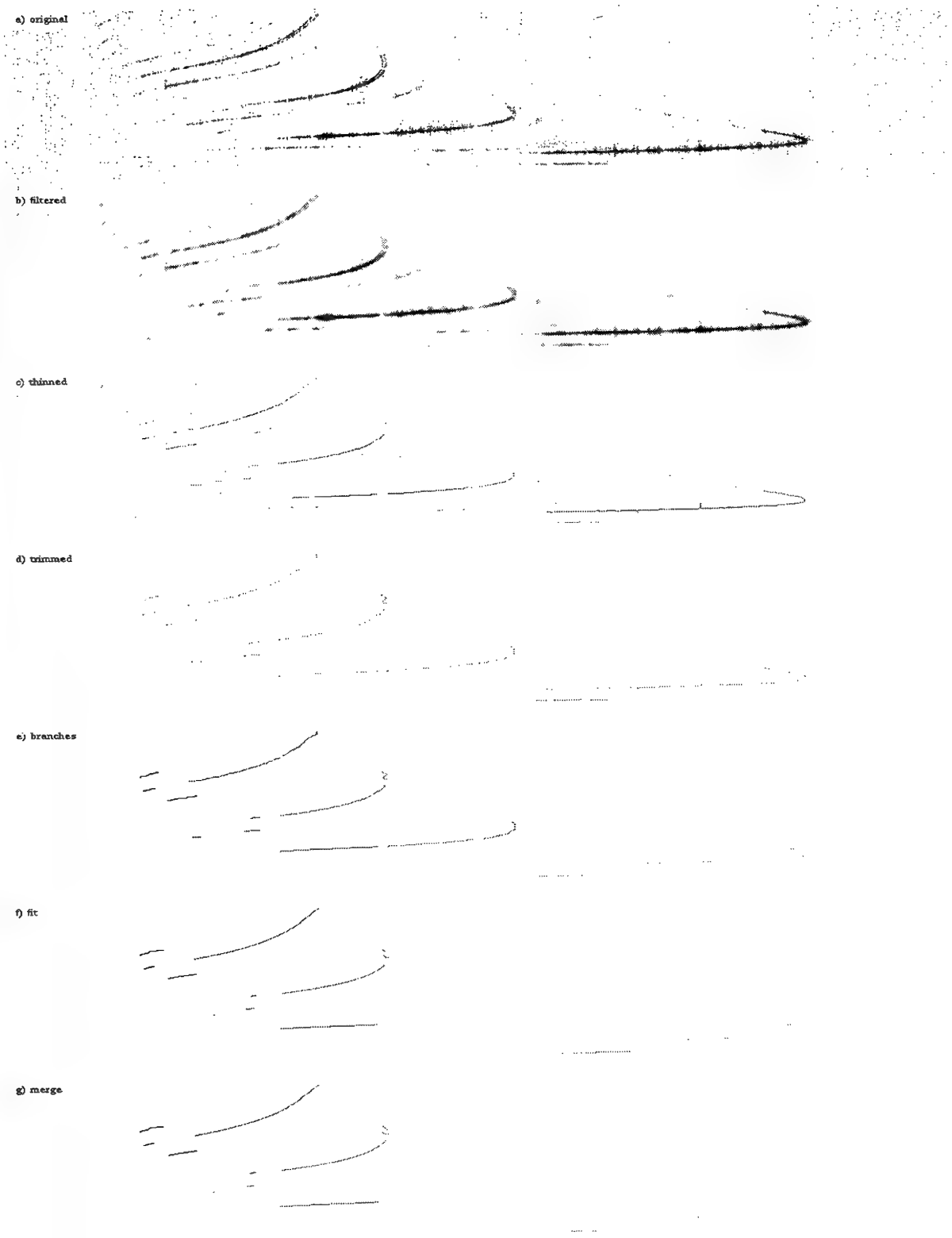


Figure 7.24: Assessment Ionogram 24.

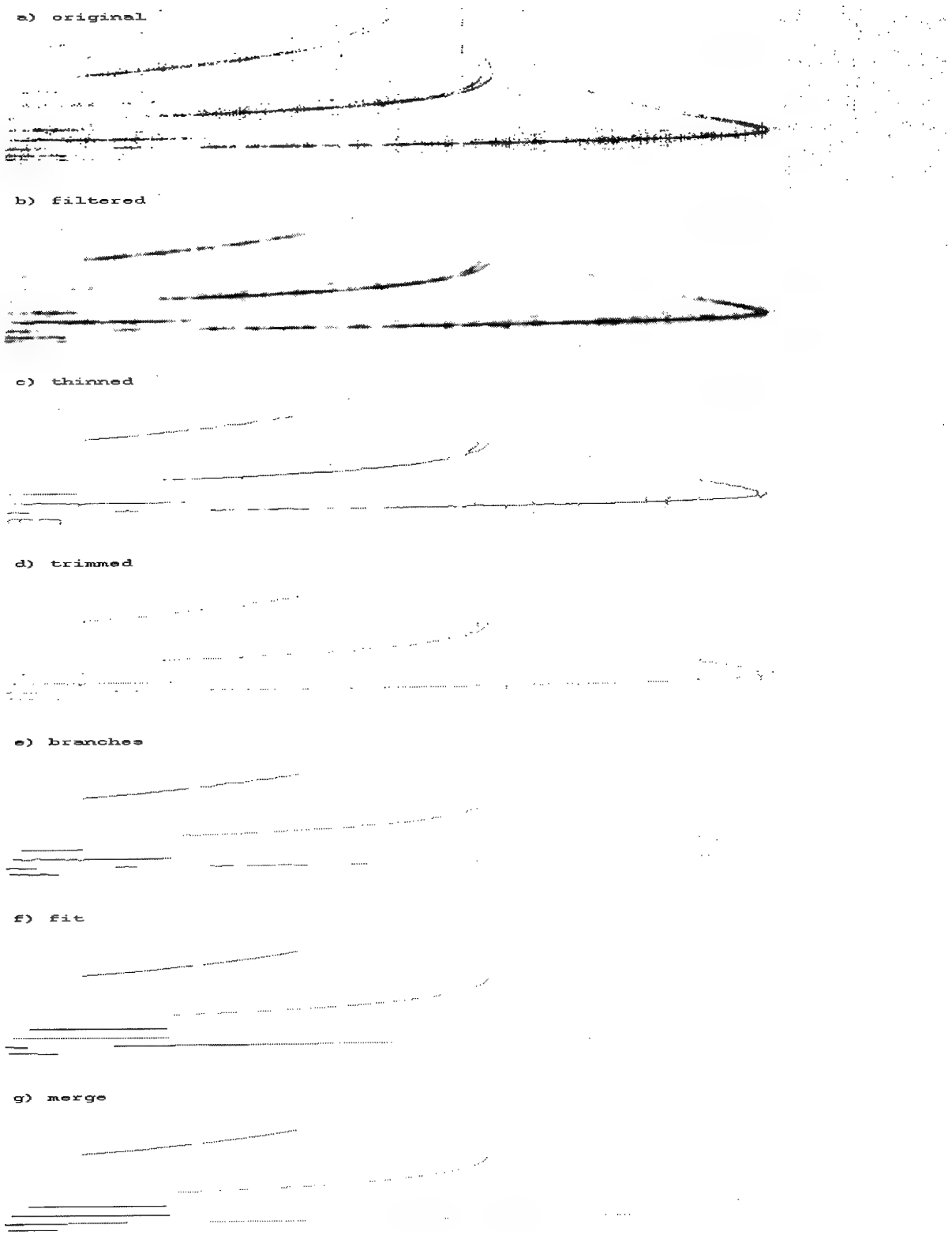


Figure 7.25: Assessment Ionogram 25.

Acknowledgements

I wish to thank Garry Newsam for his valuable suggestions and comments, and Bob Whatmough for his contribution to the algorithms. Furthermore, I wish to thank David Kettler for his interest and willingness to take the work further, Clint Wright and Peter Trudinger for originally getting me involved in the project, and Russell Clarke and Ken Lynn for their continued support of the work.

Bibliography

[1] C. S. Wright & P. L. Trudinger, "Automatic scaling of oblique-incidence ionograms — background, requirements and future strategies," *unpublished report* (7 December 1992).

[2] K. Konstantinides & J. R. Rasure, "The Khoros software development environment for image and signal processing," *IEEE Transactions on Image Processing* 3 (1994), 243–252.

[3] T. R. Crimmins, "Geometric filter for speckle reduction," *Applied Optics* 24 (1985), 1438–1443.

[4] T. R. Crimmins, "Geometric filter for reducing speckle," *Optical Engineering* 25 (1986), 651–654.

[5] R. J. Whatmough, *personal communication.*

[6] L. Alvarez, P.-L. Lions & J.-M. Morel, "Image selective smoothing and edge detection by nonlinear diffusion. II," *SIAM J. Numerical Analysis* 29 (1992), 845–866.

[7] K. V. Lever, "Well-conditioned recursive least-squared estimation algorithms," *Proceedings Fourth International Conference on Adaptive Control and Signal Processing*, Grenoble, France (1–3 July 1992).

[8] K. V. Lever, "Comments on "The filtering of oblique ionograms"," *personal communication* (1992).

[9] N. J. Redding, "Feature extraction in filtered oblique ionograms," DSTO Information Technology Division, ITD-94-15, 1994.

[10] M. Roughan, "A method for generalising binary skeletonisation algorithms to grey-scale images," Cooperative Research Centre for Sensor Signal and Information Processing, draft report, June 27, 1995.

[11] R. C. Gonzalez & P. Wintz, *Digital Image Processing* second edition, Addison Wesley, Reading, MA, 1987.

[12] T. Y. Zhang & C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM* 27 (1984), 236–239.

[13] S. Lee & G. Donohoe, "Khoros thinning algorithm.

[14] R. J. Whatmough, "Automatic threshold selection from a histogram using the "exponential hull"," *CVGIP: Graphical Models and Image Processing* 53 (1991), 592–600.

[15] C. Arcelli & G. Ramella, "Finding grey-skeletons by iterated pixel removal," *Iamge and Vision Computing* 13 (1995), 159–167.

[16] D. Kettler & N. J. Redding, "A trimming algorithm for thinned ionogram traces," *(in preparation)* (1996).

[17] M. Roughan & D. J. Percival, "Automatic extraction of traces from oblique ionograms," *(in review)* (1996).

[18] V. F. Leavers, *Shape Detection in Computer Vision Using the Hough Transform*, Springer-Verlag, London, 1993.

[19] M. Werman & Z. Geyzel, "Fitting a second degree curve in the presence of error," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17 (1995), 207–211.

[20] K. Kanatani, "Statistical bias of conic fitting and renormalization," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16 (1994), 320–326.

[21] P. T. Boggs, R. H. Byrd & R. B. Schnabel, "A stable and efficient algorithm for nonlinear orthogonal distance regression," *SIAM Journal of Scientific and Statistical Computing* 8 (1987), 1052–1078.

[22] P. T. Boggs, R. H. Byrd, J. R. Donaldson & R. B. Schnabel, "Algorithm 676 — ODRPACK: Software for weighted orthogonal distance regression," *ACM Transactions on Mathematical Software* 15 (1989), 348–364.

[23] P. T. Boggs, R. H. Byrd, J. E. Rogers & R. B. Schnabel, "User's reference guide for ODRPACK version 2.01 software for weighted orthogonal distance regression," Applied and Computational Mathematics Division, National Institute of Standards and Technology, U.S. Department of Commerce, NISTIR 92-4834, 1992.

[24] X. Cao, N. Shrikhande & G. Hu, "Approximate orthogonal distance regression method for fitting quadratic surfaces to range data," *Pattern Recognition Letters* 15 (1994), 781–796.

[25] G. Taubin, "Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13 (1991), 1115–1138.

[26] G. Taubin, "An improved algorithm for algebraic curve and surface fitting," *Proceedings Fourth International Conference on Computer Vision*, Berlin, Germany (May, 1993).

[27] M. J. D. Powell, *Approximation Theory and Methods*, Cambridge University Press, Cambridge, United Kingdom, 1981.

[28] M. J. D. Powell, *personal communication* (July, 1995).

[29] M. Gulliksson & I. Söderkvist, "Surface fitting and parameter estimation with nonlinear least squares," *Optimization Methods and Software* 5 (1995), 247–269.

[30] G. N. Newsam & N. J. Redding, "A second order test for merging model fits," *(in preparation)* (1996).

[31] K. J. W. Lynn, "Ionogram range conversion," *unpublished note* (21 May 1992).

The Autoscaling of Oblique Ionograms

Nicholas J. Redding

(DSTO-RR-0074)

DISTRIBUTION LIST

	Number of Copies
Defence Science and Technology Organisation	
Chief Defence Scientist and members of the)	1 shared copy
DSTO Central Office Executive)	for circulation
Counsellor, Defence Science, London	(Document Control sheet)
Counsellor, Defence Science, Washington	(Document Control sheet)
Senior Defence Scientific Adviser)	1 shared copy
Scientific Adviser - POLCOM)	
Assistant Secretary Science Industry Interaction	1
Director, Aeronautical & Maritime Research Laboratory	1
Navy Scientific Adviser (NSA)	1
Scientific Adviser, Army (SA-A)	1
Air Force Scientific Adviser (AFSA)	1
Electronics and Surveillance Research Laboratory	
Chief Information Technology Division	1
Research Leader Command & Control and Intelligence Systems	1
Research Leader Command, Control and Communications	1
Executive Officer, Information Technology Division	(Document Control sheet)
Manager, Human Systems Integration Group	(Document Control sheet)
Head, Software Engineering Group	(Document Control sheet)
Head, Trusted Computer Systems Group	(Document Control sheet)
Head, Advanced Computer Capabilities Group	(Document Control sheet)
Head, Intelligence Systems Group	(Document Control sheet)
Head, Systems Simulation and Assessment Group	(Document Control sheet)
Head, Exercise Analysis Group	(Document Control sheet)
Head, C3I Systems Engineering Group	(Document Control sheet)
Head, Computer Systems Architecture Group	(Document Control sheet)
Head Command Support Systems Group	1
Head Information Management Group	1
Dr Nicholas Redding	5
Publications and Publicity Officer, ITD	1
Chief Communications Division	1
Research Leader Secure Communications	1
Director Defence Communications Research Centre	1
Head Spatial Processing	1
Dr Ken Lynn	1

Mr David Kettler	1
Chief High Frequency Radar Division	1
Research Leader High Frequency Radar	1
Head Ionospheric Effects	1
Dr D. John Percival	1
Libraries and Information Services	
Defence Central Library - Technical Reports Centre	1
Manager Document Exchange Centre (MDEC) (for retention)	1
Additional copies which are to be sent through MDEC	
DIS for distribution:	
National Technical Information Centre. United States	2
Defence Research Information Centre, United Kingdom	2
Director Scientific Information Services, Canada	1
Ministry of Defence, New Zealand	1
National Library of Australia	1
Defence Science and Technology Organisation Salisbury, Research Library	2
Library Defence Signals Directorate Canberra	1
AGPS	1
British Library Document Supply Centre	1
Parliamentary Library of South Australia	1
The State Library of South Australia	1
Spares	
Defence Science and Technology Organisation Salisbury, Research Library	6

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
DOCUMENT CONTROL DATA

1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)
N/A

2. TITLE

The Autoscaling of Oblique Ionograms

3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)

Document (U)
Title (U)
Abstract (U)

4. AUTHOR(S)

Nicholas Redding

5. CORPORATE AUTHOR

Electronics and Surveillance Research Laboratory
PO Box 1500
Salisbury SA 5108

6a. DSTO NUMBER
DSTO-RR-0074

6b. AR NUMBER
AR-009-624

6c. TYPE OF REPORT
Research Report

7. DOCUMENT DATE
February 1996

8. FILE NUMBER
C9505/010/0039

9. TASK NUMBER
ADF 93/086

10. TASK SPONSOR
RTFM for
HF Communications

11. NO. OF PAGES
143

12. NO. OF REFERENCES
23

13. DOWNGRADING/DELIMITING INSTRUCTIONS

N/A

14. RELEASE AUTHORITY

Chief, Information Technology Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

APPROVED FOR PUBLIC RELEASE

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE CENTRE, DIS NETWORK OFFICE, DEPT OF DEFENCE, CAMPBELL PARK OFFICES, CANBERRA ACT 2600

16. DELIBERATE ANNOUNCEMENT
No limitation

17. CASUAL ANNOUNCEMENT No limitation

18. DEFTEST DESCRIPTORS
Oblique ionograms*
Autoscaling*
Image processing

This report details the work done on developing an autoscaling system for the Low-Latitude Ionospheric Sounding Program (LLISP). The report firstly describes the performance of a number of different filtering routines for automatic cleaning of LLISP oblique ionograms. Secondly, it then presents techniques developed to automatically extract a feature vector from a filtered ionogram. A third stage uses the feature vectors to identify the prominent modes and other important features of the ionograms, and to track these modes and features as they evolve across sequences of ionograms. This work will automate the task of identifying and tracking ionospheric propagation modes in ionograms. The resulting system will be used to process current LLISP data streams to extract large volumes of significant information: this will help the understanding of ionospheric processes and can be used as input to automatic frequency management systems for communications networks.